Karl Erich Wolff
Dmitry E. Palchunov
Nikolay G. Zagoruiko
Urs Andelfinger (Eds.)

LNAI 6581

Knowledge Processing and Data Analysis

First International Conference, KONT 2007
Novosibirsk, Russia, September 2007 and
First International Conference, KPP 2007
Darmstadt, Germany, September 2007
Revised Selected Papers

Lecture Notes in Artificial Intelligence 6581

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

Karl Erich Wolff Dmitry E. Palchunov
Nikolay G. Zagoruiko Urs Andelfinger (Eds.)

Knowledge Processing and Data Analysis

First International Conference, KONT 2007
Novosibirsk, Russia, September 14-16, 2007
and First International Conference, KPP 2007
Darmstadt, Germany, September 28-30, 2007
Revised Selected Papers

Springer

Volume Editors

Karl Erich Wolff
Hochschule Darmstadt, Germany
E-mail: karl.erich.wolff@t-online.de

Dmitry E. Palchunov
Siberian Branch of the Russian Academy of Sciences, Novosibirsk, Russia
E-mail: palch@math.nsc.ru

Nikolay G. Zagoruiko
Siberian Branch of the Russian Academy of Sciences, Novosibirsk, Russia
E-mail: zag@math.nsc.ru

Urs Andelfinger
Hochschule Darmstadt, Germany
E-mail: u.andelfinger@fbi.h-da.de

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-22139-2 e-ISBN 978-3-642-22140-8
DOI 10.1007/978-3-642-22140-8
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011930320

CR Subject Classification (1998): I.2.4, I.2, H.2.8, F.4.1, H.3.1-3

LNCS Sublibrary: SL 7 – Artificial Intelligence

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume collects the proceedings of two related international conferences on foundations and practical applications of mathematical methods of data analysis, of Formal Concept Analysis and of methods for information extraction from natural language texts. The first conference, named Knowledge - Ontology - Theory 2007 (KONT 2007), was held during September 14–16, 2007 in Novosibirsk (Russia) at the Sobolev Institute of Mathematics in cooperation with the Russian Foundation for Basic Research and the Association for Pattern Recognition and Image Analysis of the Russian Federation. The second conference, the International Conference on Knowledge Processing in Practice (KPP 2007), was held during September 28–30, 2007 in Darmstadt (Germany) at the University of Applied Sciences in cooperation with the Ernst Schröder Center for Conceptual Knowledge Processing and the Darmstadt University of Technology.

The aim of both conferences was to bring together practitioners and researchers in the interdisciplinary field of mathematical and concept-based knowledge processing. Knowledge processing today spans a broad spectrum of approaches and techniques from data analysis and pattern recognition over artificial intelligence with information retrieval and machine learning to conceptual knowledge processing with its graphical tools for knowledge visualization. From a lifecycle perspective, the field of knowledge processing covers the following main stages: data collection and pre-processing, discovery of regularities, creation of subject domain theories, application of formal knowledge structures, formal reasoning and interpretation in the application domain.

At both conferences, particular emphasis was placed on the technology and experience transfer aspects between practitioners and academic researchers in order to foster mutual learning and application-oriented research and development in the area of knowledge processing based on real empirical needs.

The contributions were all refereed and the accepted papers are collected in this volume. They cover four main focus areas which should, however, not be understood as mutually exclusive. Rather, all contributions share the common foundation that conceptual structures are essential to semantically meaningful and valid representation and processing of formal knowledge structures. The main focus areas are as follows: I: Applications of Conceptual Structures, II: Concept-Based Software, III: Ontologies as Conceptual Structures and IV: Data Analysis.

Part I: Applications of Conceptual Structures

This part on applications of conceptual structures brings together contributions that cover the history of conceptual knowledge processing (Wille) and conceptual extensions of Rough Set Theory (Ganter) as well as contributions from

modal logics (Shilov, Garanina) and applications of Temporal Concept Analysis (Wolff) for gene expression data of arthritic patients (Wollbold et al.) and information retrieval in science (Ponomaryov et al.). In most contributions, practical applications and experiences are also shown.

R. Wille outlines the vast experience base and major development steps that have been achieved since 1980 in the Darmstadt Research Group on Concept Analysis. The focus is on the application side and on historically reconstructing the evolutionary path of extending the mathematical theory of Formal Concept Analysis by a collection of methods for conceptual knowledge processing.

B. Ganter presents an application of Formal Concept Analysis in Rough Set Theory, introduced by Pawlak (1982), who based this theory on the idea of an approximation of a set by equivalence classes of an equivalence relation, called indiscernibility relation, on a given universe U of objects. Ganter generalizes this idea by introducing a preorder (reflexive and transitive) on the set U. The conceptual representation shows that this preorder expresses that one object concept is a subconcept of another one. The corresponding definable sets are the preorder ideals. They form a distributive lattice as in the case of the indiscernibility relation. Connections to functional dependencies, linguistic variables, and decision making are mentioned.

S.O. Kuznetsov presents an innovative algorithm which constructs the lattice of graph sets in a top-down way, starting from the smallest subgraphs of the graphs in a dataset. In contrast to previous bottom-up algorithms this approach allows one to stop at an appropriate level of approximation (projection), thus saving much time and space and making the computational complexity more tractable.

N.V. Shilov and N. Garanina present a summary of recent studies of the model-checking problem for certain combinations of propositional logics for knowledge, time, and actions. After a short overview of these logics they provide a brief introduction to modal logics and Kripke models using the example of Elementary Propositional Dynamic Logic (EPDL). Then they describe propositional logics for epistemic agents, branching temporal logic with actions, combined logics of knowledge, actions, and time, and finally model checking problems in combined logics.

K.E. Wolff presents new results in Temporal Concept Analysis. He starts with an example of a moving high-pressure zone and explains the basic notions in Temporal Conceptual Semantic Systems referring to this example. One of the main results is the purely conceptual introduction of the notion of a state of an object at some time granule (with respect to some view and some selection). The states are special cases of traces of an object. These traces generalize the notion of a volume of an object in some space. In practical applications the used semantic scales are a valuable tool for the representation of a suitable granularity. This is shown by an application in the chemical industry setting where the behavior of a distillation column over 20 days is represented by a lifetrack in a nested line diagram visualizing four many-valued attributes. The same technique is also used in the next article in the field of biomedicine.

J. Wollbold, R. Huber, R.W. Kinne, and K.E. Wolff present a cooperation between scientists from medicine, biology, and mathematics. They investigate disease processes of rheumatoid arthritis using time series of gene expression data. For the purpose of understanding their complicated temporal data they represent these data as a Temporal Conceptual Semantic System as introduced by Wolff (in the previous article). The application of transition diagrams which represent simultaneously the movement of several patients in some genetic space, represented by a concept lattice, yields valuable insight into genetic processes, for example, by finding new hypotheses concerning gene regulation. By applying concept-based analysis techniques the article points to innovative directions for research compared with the current state of the art in this field of biomedicine.

D. Ponomaryov, N. Omelianchuk, V. Mironova, E. Zalevsky, N. Podkolodny, E. Mjolsness, and N. Kolchanov present an innovative approach to automate the extraction process of formal knowledge structures from scientific publications. The goal is to contribute to a better exploitation and more comprehensive usage of already published research results in a specific scientific domain. The value of this work lies in accelerating the formation of hypotheses and theories, which is demonstrated with a concrete example in the field of botany.

Part II: Concept-Based Software

This part on concept-based software focuses around methodical challenges and approaches for supporting software development and software usage with concept-based tools and techniques.

A.S. Kleshchev explores the potential of ontologies as conceptual structures for supporting domain analysis and simulation in software development. He extends the approach also to technical applications, e.g., in optimizing compilers based on ontologies in formal knowledge structures.

G. Stumme discusses problems arising from the empirical success of social bookmarking. While social bookmarking allows for a bottom-up approach in jointly organizing and sharing knowledge assets and bookmarks on the Internet, users are increasingly getting lost in the tons of available meta-information that they are collectively generating. The author presents recent empirical research results that might help users in keeping control over the mass of poorly structured information elements in social bookmarking systems.

U. Priss and J. Old explore new ways of reducing large formal data structures in the field of lexical databases so that they are better understandable for humans. The challenge here is to keep as much as possible of all the relevant semantics of a specific data search while omitting as much as possible of the non-relevant or empty search results. To this purpose they borrow the term ‘weeding’ from biology: with every search they want to preserve as much of the valuable ingredients while collecting as few weeds as possible. They also present practical implementations for their ideas.

Part III: Ontologies as Conceptual Structures

This part on ontologies as conceptual structures focuses on developing taxonomies and conceptual structures, which are increasingly called 'ontologies,' for given problem domains. The aim is to support meaningful knowledge representation, knowledge communication and knowledge retrieval.

D.E. Palchunov describes the idea of a 'Virtual Catalog,' a synthesis of search engines and Internet catalogs. A Virtual Catalog can be considered as an ontology-based technology for information retrieval. To formulate and satisfy the information need of the users, three types of ontologies are created: an ontology of the subject domain of interest, an ontology for various types of Internet resources, and an ontology for the types of information search tasks (which is not yet implemented in the described system). The elaboration of the toolkit for the formalization of search queries is a very important part of this research. For that purpose, ontologies are developed as networks of sentences of First-Order-Logic. Two application fields are used to demonstrate the practicality of the approach.

I.L. Artemieva presents experiences and problems with the construction of ontologies for domains with complicated structure—as for example a multilevel ontology for chemistry, which serves as the main example in this paper to introduce a general method for constructing multilevel ontologies. This paper is part of a project of the Presidium of the Russian Academy of Sciences.

Y.A. Zagorulko and O.I. Borovikova describe a comprehensive approach to formalize knowledge structures in the field of humanities using ontologies. They also describe a method for building such an ontology and an ontology description logic, and they have developed an end-user-oriented editor to actively contribute to the further evolution of the ontology. The resulting IT system is accessible via a so-called knowledge portal.

A. Marchuk presents a dynamic information system on historical facts. The unique characteristic for historical facts is that a given topic (information about countries, persons, office documents) evolves over time. What is a historic fact today, is very often a different historic fact tomorrow as new developments have to be added to the fact to keep it up to date. The proposed approach is capable of automatically keep track of dynamic changes to historic facts. The benefit is that we can thus create databases which no longer become obsolete. The approach has been tested in practical applications at the A.P. Ershov Institute of Informatics Systems.

N. Loukachevitch discusses problems that arise when developing ontologies in the field of linguistics and natural languages. In (natural) languages taxonomic relationships cannot always be resolved in simple logical structures. For example, the basic rule "If class A is a subclass of class B, then each instance of class A is also an instance of B" may not be true in many cases for the conceptual structures in natural languages. The paper addresses ways for revealing typical mistaken taxonomic relationships in such situations. The findings have been derived from practical experiences with a large thesaurus and ontology on natural sciences and technologies.

Part IV: Data Analysis

This part on data analysis focuses on mathematical considerations, techniques, and algorithms for automated discovery of knowledge structures, data mining, similarity analysis, and approaches to visualize knowledge structures.

N.G. Zagoruiko introduces a formal representation of an empirical theory and applies it for tasks of several types in data mining, for example, the task of generating a data-dependent partition of a given set. Then the corresponding formal empirical theory represents (all) possible clustering methods as its formal objects, which are formally described in some language V by a set of characteristics (or attributes). Then a test algorithm T constructs for each clustering method a quality value. With respect to some chosen threshold the class of admissible clustering methods is introduced as the set of those clustering methods whose quality value exceeds the chosen threshold. In the second part of the paper the author reports on his approach for the construction of a measure of similarity, namely, his successful Function of Rival Similarity (FRiS). The paper finishes with some ideas concerning a strengthening of the empirical Data-Mining Theory.

I.A. Borisova, V.V. Dyubanov, N.G. Zagoruiko, and O.A. Kutnenko presents practical applications of the Function of Rival Similarity (FriS) to basic tasks of data mining. They first demonstrate the use of the FRiS-function for selecting typical representatives (stolps) of classes, which will be used for the recognition of new objects. Then, they demonstrate the application of FRiS to several tasks, for example, the tasks of type SDX, i.e., to the simultaneous construction of a classification (task S) of observable objects, building decision rules (task D) and the selection of informative subsets of attributes (task X). It is also shown that the FriS-based algorithms are invariant to the ratio of the number of objects to the number of attributes in the dataset and to the type of the probability distribution of the samples.

V.B. Barakhnin, V.A. Nekhayeva, and A.M. Fedotov analyze and compare three algorithms to solve the problem of computerized selection of Internet documents on scientific subjects based on similarity determination. It is concluded that the FRiS-algorithm which is based on Zagoruiko's principle of rival similarity is the best out of the analyzed algorithms, followed by the greedy algorithm.

E. Vityaev and S. Smerdov consider predictions provided by Inductive Statistical (I-S) inference which are, by Hempel, statistically ambiguous. To avoid this ambiguity, Hempel introduced the Requirement of Maximum Specificity (RMS). Vityaev and Smerdov introduce maximum specific (MS) rules and prove that they satisfy the RMS. The authors also prove that I-S inferences using MS rules avoid the problem of statistical ambiguity. To introduce a probability of events and sentences they use the product probability as in the case of statistical independence; using this product probability they introduce the notion of a probabilistic law and study semantic probabilistic inference, probabilistic Herbrand models, and predictions based on semantic probabilistic inference. Finally, they shortly describe a computer program, DISCOVERY, based on semantic probabilistic inference.

B. Kovalerchuk and A. Balinsky propose a novel approach for analyzing multidimensional data. They focus on the observation that often a human user can easily catch a border between patterns visually, but its analytical form can be quite complex to describe and difficult to discover by formal algorithms. The paper describes a new technique for extracting patterns and relations visually from multidimensional data using the process of data monotonization that gives the opportunity to use the theory of monotone Boolean and k-valued functions. The major novelty of the approach is in visualization of structural relations between n-dimensional objects instead of traditional attempts to visualize each attribute value of n-dimensional objects. Experiments with breast cancer data show the advantages of this approach in uncovering a visual border between benign and malignant cases in breast cancer diagnostics.

G. Lbov and V. Berikov present an algorithm for event tree construction on the basis of an analysis of expert knowledge and multivariate time series. The algorithm uses the Bayesian criterion for decision tree pruning. It can be used for the analysis of extreme events in the conditions of high a priori uncertainty about them. In this case an expert can contribute additional statistical information concerning the object under investigation. Experiments with artificial and real data sets confirm the usefulness of the proposed algorithm.

November 2010

Dmitry E. Palchunov
Nikolay G. Zagoruiko
Karl Erich Wolff
Urs Andelfinger

KONT 2007 Organization

Executive Committee

Conference Committee	
Conference Chair	Y.I. Zhuravlev, Full Member of the Russian Academy of Sciences
Co-chairs	D.E. Palchunov and N.G. Zagoruiko
Program Committee	Ablamejko S.V. (Minsk, Belarus) Cheremisina E.N. (Dubna, Russia) Fedotov A.M. (Novosibirsk, Russia) Gavrilova T.A. (St. Petersburg, Russia) Gladun V.P. (Kiev, Ukraine) Ganter B. (Dresden, Germany) Herre H. (Leipzig, Germany) Kleshchev A.S. (Vladivostok, Russia) Kolchanov N.A. (Novosibirsk, Russia) Kovalerchuk B. (Seattle, USA) Kuznetsov S.O. (Moscow, Russia) Lbov G.S. (Novosibirsk, Russia) Marchuk A.G. (Novosibirsk, Russia) Mihalskij R.S. (Washington, USA) Samokhvalov K.F. (Novosibirsk, Russia) Sviridenko D.I. (Moscow, Russia) Tselishchev V.V. (Novosibirsk, Russia) Vasiljev S.N. (Moscow, Russia) Vityaev E.E. (Novosibirsk, Russia) Wolff K.E. (Darmstadt, Germany) Zelger J. (Innsbruck, Austria)
Organizing Committee	Palchunov D.E., Chair Zagoruiko N.G., Chair Yakhyaeva G.E., Scientific Secretary Borisova I.A . Korovina M.V. Lbov G.S. Pavlovskij E.N. Rjaskin A.N. Salomatina N.V. Vasilenko N.M. Vityaev E.E. Vlasov D.J.

Sponsoring Institutions

Russian Fund of Basic Research
Russian Academy of Natural Sciences
Institute of Mathematics of the Siberian Branch of the
Russian Academy of Sciences
Novosibirsk State University

KPP 2007 Organization

Executive Committee

Conference Committee	
Honorary Chair	Rudolf Wille University of Technology, Darmstadt, Germany
Conference Chair	Karl Erich Wolff University of Applied Sciences, Darmstadt, Germany
Co-chair	Urs Andelfinger University of Applied Sciences, Darmstadt, Germany
Program Committee	Bernhard Ganter, University of Technology, Dresden, Germany Wolfgang Hesse, University of Marburg, Germany Sergei O. Kuznetsov, Higher School of Economics, Moscow, Russia Dmitry E. Palchunov, Institute of Mathematics SB RAS, Novosibirsk State University, Novosibirsk, Russia Nikolay V. Shilov, Institute of Informatics Systems, Novosibirsk, Russia Uta Störl, University of Applied Sciences, Darmstadt, Germany Gerd Stumme, University of Kassel, Germany
Organizing Committee	Urs Andelfinger Karl Erich Wolff

Sponsoring Institutions

Darmstadt University of Applied Sciences
Center for Research and Development at
Darmstadt University of Applied Sciences
Mathematics and Science Faculty at Darmstadt University of Applied Sciences
Computer Science Faculty at Darmstadt University of Applied Sciences
Ernst Schröder Center for Conceptual Knowledge Processing

Table of Contents

Part I: Applications of Conceptual Structures

Part II: Concept-Based Software

Part III: Ontologies as Conceptual Structures

Part IV: Data Analysis

Conceptual Knowledge Processing: Theory and Practice*

Rudolf Wille

Fachbereich Mathematik, Technische Universität Darmstadt,
Schloßgartenstr. 7, D–64289 Darmstadt
wille@mathematik.tu-darmstadt.de

Abstract. *Conceptual Knowledge Processing* is understood as the general scientific discipline which activitates acts of thinking such as representing, reasoning, acquiring, and communicating conceptual knowledge. In this contribution, first a short introduction to the foundation of Conceptual Knowledge Processing is presented. Then the theoretical background based on *Formal Concept Analysis* is sketched. The main content of this contribution is a discussion of the *practice of Conceptual Knowledge Processing* which demonstrates for twelve selected general acts of thinking how they can be successfully applied by using different methods of Conceptual Knowledge Processing.

Contents

1 Conceptual Knowledge Processing

The term *"Conceptual Knowledge Processing"* denominates a central idea which brings together humanistic and social scientists, mathematicians, and computer

* This paper is an updated English version of the German publication [Wi00b].

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 1–25, 2011.

scientists to support a humanitarian approach in dealing with the means of information processing, transfer, and knowledge communication. The shared objectives are those methods and instruments that process information and knowledge which support human beings to think, to judge and act rationally and to create the circumstances for critical discourse. The organization, established to promote the request of Conceptual Knowledge Processing, is the "ERNST-SCHRÖDERZENTRUM *für Begriffliche Wissensverarbeitung e.V.*" founded in Darmstadt in 1993 [Wi94].

Conceptual Knowledge Processing prefers an understanding of knowledge according to which *ambitious knowledge* can only be obtained and supported through conscious reflexion, discursive argumentation, and human communication based on the existent understanding of life and environment, social conventions and personal experience [Wi00a]. This vision of knowledge strictly related to humans accords with the present ideas of knowledge management stating "that knowledge (in contrast to data and information) is always linked to people" [PRR99]. The connection between data, information, and knowledge may therefore, according to K. Devlin [De99], be defined as follows:

$$\begin{aligned} \text{data} &= \text{signs} + \text{syntax} \\ \text{information} &= \text{data} + \text{meaning} \\ \text{knowledge} &= \text{internalized information} + \text{ability to utilize it} \end{aligned}$$

The component "conceptual" in the name "conceptual knowledge processing" is to stress the constitutive role of the thinking, debating and communicating human being for knowledge and its processing.

"Processing" in Conceptual Knowledge Processing refers to the process in which something is gained by knowledge which is knowledge again or something approximating knowledge such as a forecast, an opinion, a casual reason etc. Since for processing knowledge formal procedures and means of language are used which causes formal representations of knowledge, humans have to constitute again knowledge out of such representations.

To understand this process, the basic relation between the *form* and the *content* must be clarified for Conceptual Knowledge Processing. A branch of philosophy which makes a statement of principle on this is the *pragmatic philosophy* which was initiated by Ch. S. Peirce and is presently continued among others in the discourse philosophy of K.-O. Apel and J. Habermas. According to pragmatic philosophy, knowledge is formed in an unlimited process of human thinking, debating and communicating. In this process, reflection on the effects of thinking is significant and real experiences lead to re-thinking time and again. In this process, form and content are related to the extent that they may not be separated without loss (cf. [Wi94]).

2 Theory

The theory of Conceptual Knowledge Processing is mainly based upon formalizations of the traditional philosophical logic with its doctrines of concept, judgment,

and conclusion. The mathematical theory of concepts and concept hierarchies has been elaborated to a great extent in the last thirty years mostly at the TU Darmstadt under the name *"Formal Concept Analysis"* [GW99]. In the last ten years, Formal Concept Analysis has been extended to a mathematical theory of judgment and conclusion called *"Contextual Logic"* [Wi08], which was based on a mathematization of J. F. Sowa's theory of conceptual graphs [So84]. With all these formalizations, contents are integrated in such a way that the arising structures can effectively support the process of knowledge creation.

To comprehend this, the fundamental terms of Formal Concept Analysis shall be first explained via an example. A data table as in Fig. 1 is mathematically

	age of entering office <60	age of entering office >=60	one office period	two office periods	CDU	SPD	FDP
Heuss		×		×			×
Lübke		×		×	×		
Heinemann		×	×			×	
Scheel	×		×				×
Carstens		×	×		×		
Weizsäcker		×		×	×		
Herzog		×	×		×		
Rau		×	×			×	

Fig. 1. Formal context of the German presidents from 1949 to 2004

understood to be a *formal context* consisting of a set of *objects* (in the example: the Presidents of the Federal Republic of Germany up to 2004), a set of *attributes* (in the example: age of entering office < 60, age of entering office ≥ 60, one office period, two office periods, party membership: CDU, SPD, or FDP), and a *relation* (in the example represented by the crosses in the data table) indicating which object has which attribute.

Each formal context gives rise to formal concepts which form according to the subconcept-superconcept ordering the so-called *concept lattice* of the context. A *formal concept* consists of a set of objects, its extent, and a set of attributes, its intent. The extent consists of all those objects of the context which have all

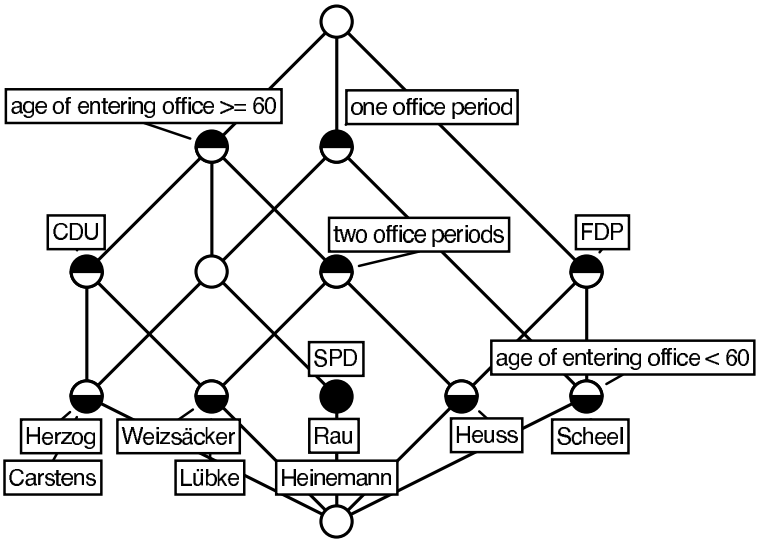


Fig. 2. Concept lattice of the German presidents from 1949 to 2004

attributes of the intent and, dually, the intent consists of all those attributes of the context which apply to all objects of the extent. A formal concept is a subconcept of another concept if its extent is part of the extent of the other concept or - which is equivalent - if its intent contains the intent of the other concept.

To make conceptual connections in data tables more transparent, it has proven effective to represent the concept lattice derived from such a data table by a line diagram as shown in Fig. 2. The small circles of the line diagram represent the formal concepts of the given context and the ascending sequences of line sections represent subconcept-superconcept ordering. Thus, the small circle labelled by "Heuss" depicts a subconcept of the concept which is represented by the small circle labelled by "FDP". This indicates that Heuss was a member of the FDP party.

In general, the extent and the intent of a formal concept may be read from a line diagram as follows: The extent of a concept consists of all objects whose denominations are attached to a small circle belonging to a descending sequence of line segments starting from the small circle representing the considered concept. Dually, the intent of a concept consists of all attributes whose denominations are attached to a small circle belonging to an ascending sequence of line segments starting from the small circle representing the considered concept. Hence, in Fig. 2, the small circle without labels on the right of the CDU circle represents the formal concept whose extent consists of the German presidents "Carstens", "Herzog", "Heinemann" and "Rau" and whose intent consists of the attributes "age of entering office ≥ 60" and "one office period".

From what was said it follows that the original context can be reconstructed from the line diagram, i. e. no data is lost when a concept lattice and its line diagram is built up. For arbitrary data tables informative concept lattices can be constructed, even when these data tables contain, instead of crosses, numbers or other arbitrary symbols; then the method of *conceptual scaling* [GW99] is used which allows to construct formal contexts with corresponding concept lattices out of data tables and even databases. To reveal the multitude of possible relations, *relational scaling* [GW99] has been introduced to contextual logic which allows, for instance, the derivation of linguistic formulations and their representation by concept graphs. Since the early 1980's software for conceptual analysis of data has been developed to a great extent; up to now, TOSCANA is the most attractive program system for Formal Concept Analysis and its applications (see [KSVW94], [BH05]).

How Formal Concept Analysis is based on the the mathematical notions of a formal context and its concept lattice, this shall be briefly sketched at the end of this theory section. A *formal context* is defined as an incidence structure (G, M, I) where G and M are sets and I is a binary relation between G and M (i.e. $I \subseteq G \times M$); the elements of G are called *objects*, those of M *attributes*, and the relationship $(g, m) \in I$ (often described also by gIm) is read: the object g has the attribute m. A Galois connection between the power sets of G and M is defined by the following derivation operators ($X \subseteq G$ and $Y \subseteq M$):

$$X \mapsto X' := \{m \in M \mid gIm \text{ for all } g \in X\},$$
$$Y \mapsto Y' := \{g \in G \mid gIm \text{ for all } m \in Y\}.$$

A *formal concept* of a formal context $\mathbb{K} := (G, M, I)$ is a pair (A, B) with $A \subseteq G$, $B \subseteq M$, $A = B'$, and $B = A'$; A is called the *extent* and B is called the *intent* of the formal concept (A, B). The subconcept-superconcept-relation is mathematized by $(A_1, B_1) \leq (A_2, B_2) :\Leftrightarrow A_1 \subseteq A_2 (\Leftrightarrow B_1 \supseteq B_2)$. The set of all formal concepts of $\mathbb{K}$ together with the order-relation $\leq$ is denoted by $\underline{\mathfrak{B}}(\mathbb{K})$. A general method of construction of formal concepts uses the derivation operators to obtain for $X \subseteq G$ and $Y \subseteq M$ the formal concepts (X'', X') and (Y', Y''). For an object $g \in G$ its *object concept* $\gamma g := (g'', g')$ is the smallest concept in $\underline{\mathfrak{B}}(\mathbb{K})$ whose extent contains g, and for an attribute $m \in M$ its *attribute concept* $\mu m := (m', m'')$ is the largest concept in $\underline{\mathfrak{B}}(\mathbb{K})$ whose intent contains m.

Therefore, if in a line diagram of $\underline{\mathfrak{B}}(\mathbb{K})$ the object names are attached to the circles of the corresponding object concepts and the attribute names are attached to the circles of the correponding attribute names, then extent and intent of the represented concepts and therefore also the corresponding context can be read from the inscribed line diagram of the concept lattice. The following basic theorem on concept lattices shows that one can even examine by the inscribtions whether the presented line diagram represents the concept lattice of the given formal context or not.

Basic Theorem on Concept Lattices [Wi82]. *Let $\mathbb{K} := (G, M, I)$ be a formal context. Then $\underline{\mathfrak{B}}(\mathbb{K})$ is a complete lattice, called the* concept lattice *of $\mathbb{K}$, whose infima and suprema can be described as follows:*

$$\bigwedge_{t\in T}(A_t, B_t) = (\bigcap_{t\in T} A_t, (\bigcup_{t\in T} B_t)''), \quad \bigvee_{t\in T}(A_t, B_t) = ((\bigcup_{t\in T} A_t)'', \bigcap_{t\in T} B_t).$$

In general a complete lattice L is isomorphic to $\underline{\mathfrak{B}}(\mathbb{K})$ if and only if there exist mappings $\tilde{\gamma} : G \to L$ and $\tilde{\mu} : M \to L$ such that $\tilde{\gamma}G$ is $\bigvee$-dense in L (i.e. $L = \{\bigvee X \mid X \subseteq \tilde{\gamma}G\}$), $\tilde{\mu}M$ is $\bigwedge$-dense in L (i.e. $L = \{\bigwedge X \mid X \subseteq \tilde{\mu}M\}$), and $gIm \iff \tilde{\gamma}g \leq \tilde{\mu}m$ for $g \in G$ and $m \in M$; in particular, $L \cong \underline{\mathfrak{B}}(L, L, \leq)$.

3 Practice

The ways in which the theoretical basis of conceptual knowledge processing can practically prove its value shall be demonstrated using selected examples. Each example represents a given application characterized by a general act of thinking or reasoning [Wi99]. At the beginning of each example, the specific meaning of the characteristic act of thinking for Conceptual Knowledge Processing is explained. For determining those meanings relevant dictionaries were used together with word meanings which have been developed within the practice of Conceptual Knowledge Processing.

3.1 Exploring

> *"Exploring" means looking for something of which one has only a vage idea or awareness.*

For such an understanding of exploring, a typical example is looking for literature without knowing exactly what pieces of literature concerning a given theme are available. Exploring therefore involves an iterative learning about the body of knowledge before discovery can be successful. To give substance in this effort, a *TOSCANA system for exploring literature* was developed for the Library of the Center for Interdisciplinary Technology Research (ZIT) at Darmstadt University of Technology.

This system is based on a respectively actualized data context whose objects are the *books* in the ZIT-library and whose attributes are *standardized keywords*, while the relation of the context shows which keywords are assigned to which book. To explore the large concept lattice of the data context in a purpose-oriented manner, line diagrams of meaningful projections of the large concept lattice were drawn which represent 137 informative *themes*. These projections can be activated as *conceptual search structures* in arbitrary combinations. Fig. 3 shows the line diagrams for the two themes "computer science and knowledge processing" and "town and traffic"; they represent partial progress on search for a documentation aimed at literature on "expert systems in traffic". In the course

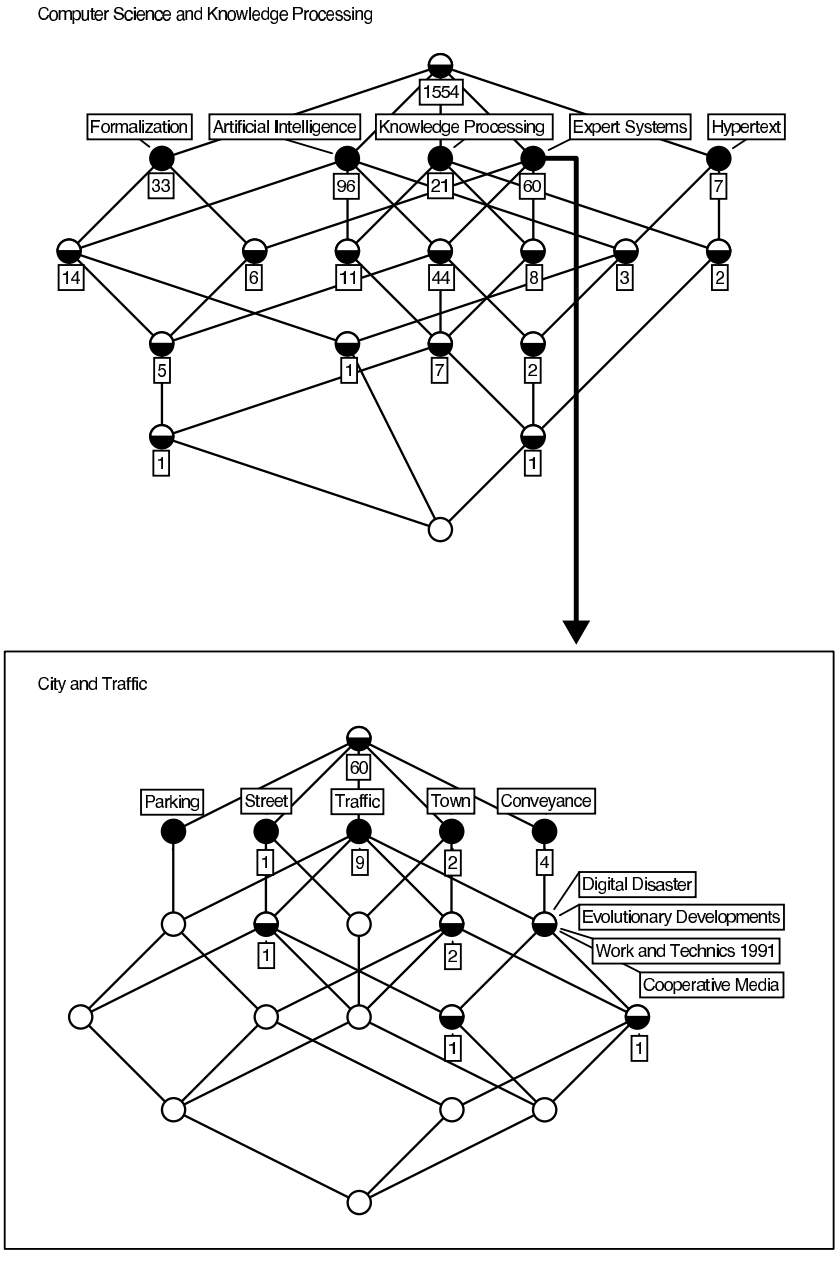


Fig. 3. Exploring literature with concept lattices about the theme "expert systems in the area of traffic"

of this exploration the theme "computer science and knowledge processing" was chosen first; then the theme "town and traffic" was zoomed into the small circle of the first theme which is labelled from above by the attribute name "expert system" and from below by the number "60". This causes that the second line diagram and the distribution of the 60 books with the keyword "expert systems" are shown in relation to the keywords of the theme "town and traffic". Subsequently, the book number marked by 4 at the small circle with upward line segments leading to the keywords "traffic" and "means of transport" was selected which caused the display of the 4 book titles "Digital Disaster", "Evolutionary Ways into the Future", "Yearbook of Work and Technology 1991" and "Cooperative Media" (detailed information can be found in [RW00]).

3.2 Searching

> *"Searching" shall be understood as looking for something which one can more or less specify but not localize.*

When using the methods of Conceptual Knowledge Processing for search, the objective is usually to find explicit knowledge whose existence is already known. In 1992, the Darmstadt Research Group on Formal Concept Analysis was confronted with such a situation when the group was asked by the "Ministry for Building and Housing" of the German State "Nordrhein-Westfalen" to develop for them a prototype of a tool which supports architects in finding the relevant laws and regulations about building constructions. In the period of three years a TOSCANA-system was developed as such a tool [KSVW94], [EKSW00]. For this system an extensive data context was elaborated

- whose objects were the relevant *paragraphs and text units* of the current laws and regulations, and
- whose attributes, which are understood as keywords, refer to the *building parts and the requirements*.

As it is characteristic for TOSCANA-systems, multiple projections of the concept lattice of the underlying data context are represented by line diagrams which can be used as conceptual query structures.

In Fig. 4, the line diagram of the search structure "shell of a building" is depicted and reflects on the keywords (attributes) "chimney", "roof", "basement floor", "wall", "fire wall", "staircase", "stairs", and "ceiling". In order to be able to elaborate search, the search structure "shell of a building" may be increased by other search structures such as "interior", "requirements" etc. But it has been experienced that even already one search structure with its conceptual relationships may support the finding of relevant text units. Furthermore, the overview given by a line diagram can enable experts to find mistakes in the basic data contexts which, during the project with the ministry, has contributed to improve the quality of the data again and again.

Fig. 4. The search structure "shell of a building" of an exploration-system for building laws and regulations

3.3 Recognizing

"Recognizing" means to discern and to clarify something.

In Conceptual Knowledge Processing, the method of discerning and clarifying something is to make conceptual relationships between form and contents obvious in multiple ways. In a research project, which the Darmstadt Research Group on Formal Concept Analysis performed in cooperation with a Swiss retail group, the task was to investigate how far the use of Formal Concept Analysis methods could clarify customers' behaviour to buy (by incorporating a TOSCANA-system into the data warehouse domain) and therewith support the database-marketing-activities of the combine [HSW00], [He00]. The results of the project were received positively, in particular, because marketing executives were able to discuss the data with their business partners based on the presentation of the data as concept lattices. New insights on the shopping habits of the customers were gained and new ideas for marketing activities were subsequently developed.

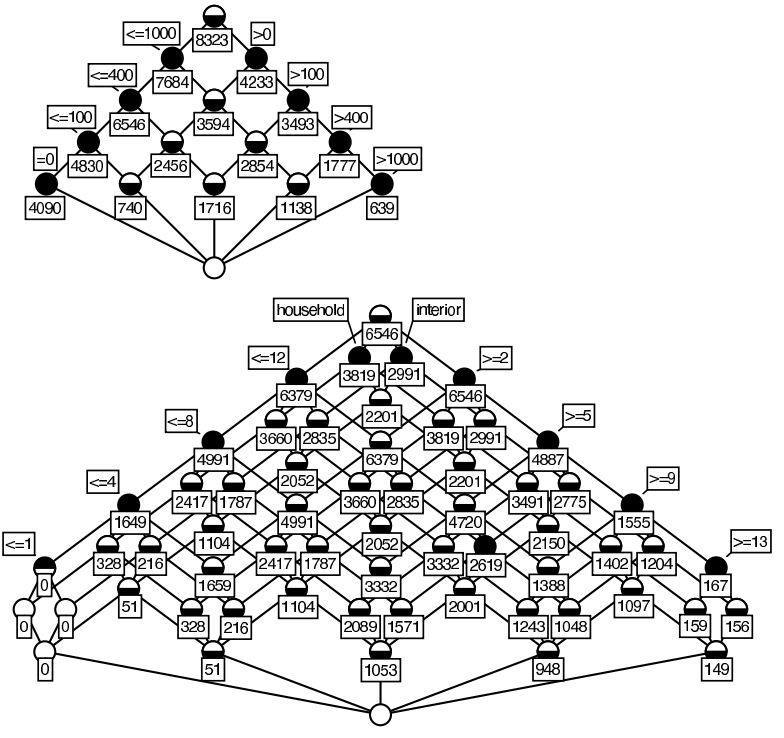


Fig. 5. Concept lattices for analyzing purchase activities (above: expenses of women clothes; below: number of the used departments and purchase of household articles and interior decorations)

In Fig. 5 the line diagram on the top shows how the spending of 8,323 female customers in "women's wear" is distributed (for example: 1,716 customers spent more than SFR 100 and at most SFR 400 and 1,777 customers spent more than SFR 400). The line diagram on the bottom combines two *aspects of the purchase activity*. The line diagram with large circles depicts the distribution according to the number of departments visited during the shopping session, the more detailed line diagram inside the larger circles further explains the respective number (in the upper part of the big circle) as a department's share "household goods" (left side), "interior decoration" (right side) and both (below). For the marketing department the diagram in steps was interesting for the reasons that follow. Of the 6,546 female customers who have spent little on clothing (SFR 400 at the most) it shows the number of departments in which they purchased goods and also their proportion of shopping in "household goods" and "interior decoration". The 2,001 customers who shopped in 5 to 12 departments, particularly "household goods" and "interior decoration", may well be chosen as a target group for *personalized mailings* to stimulate their purchasing in the department "women's wear".

3.4 Identification

> *"Identifying" shall mean to determine the taxonomic position of an object within a given classification.*

Since a taxonomic position in classifications is usually charaterized by specific attributes, the line diagrams of concept lattices are very well suited to support the identification of objects (e.g. identifying plants). For the renowned exhibition on symmetry which was held on the "Mathilden Höhe" in Darmstadt in 1986, B. Ganter and J. Richter wrote a computer program which is able to determine the symmetry types of two-dimensional ornaments (e.g. of wallpaper prints) and can further be used to generate two-dimensional ornaments that match a chosen symmetry type. The program is based on a formal context having as objects the 17 symmetry types of two-dimensional patterns and as attributes the symmetry properties which crystallographers use to identify planar symmetries [Wi87].

Fig. 6 shows the concept lattice of this symmetry context. The way that this lattice supports the *identification of a symmetry type* shall be explained using the two-dimensional ornament depicted in Fig. 6 too. In the line diagram the circles filled black indicate the steps in the identification process. It starts with the uppermost circle, i.e. the concept, which includes all 17 symmetry types in its extent. The understanding that "two non-parallel reflection axes" yield a pattern of symmetries of the ornament leads to the blackened circle two steps down; the concept depicted by this circle has only the symmetry types p6, p6m, cmm, p3m1, p31m, p4m and p4g left in its extent. Discovering a "90 degree rotation" p4m and p4g as a symmetry leads to the formal concept which has only the symmetry types left in its extent. Since the fixpoint of this rotation lies not on a reflection axis of some rotation symmetry of the presented ornament, one can conclude that the example belongs to symmetry p4g.

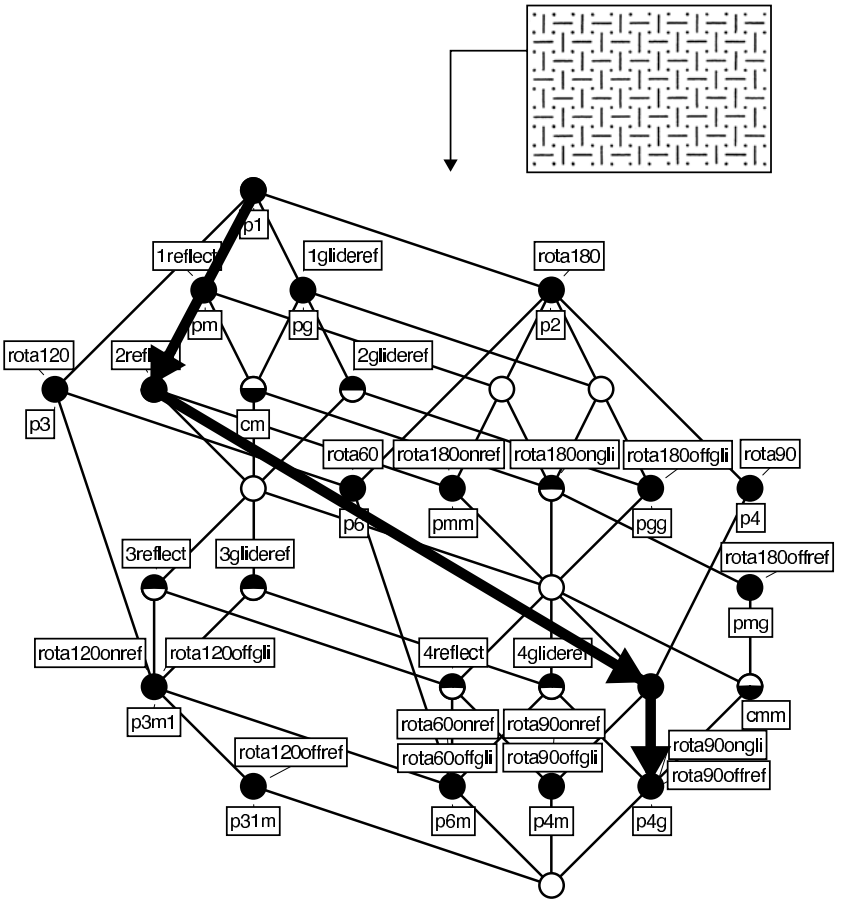


Fig. 6. Concept lattice of symmetry types of two-dimensional patterns

3.5 Investigating

"Investigating" means to study by close examination and systematic inquiry.

The formal structures of Conceptual Knowledge Processing, which make data understandable without any simplification of the original data, have proven their worth in a wide range of applications by showing (and discovering) relationships between data. In this respect the cooperation with the political scientist B. Kohler-Koch has had important outcomes. The aim of the cooperation was the analysis of a data context on *international cooperations bound by standards and regulations*. Working on this problem led to the development of TOSCANA-systems. The objects of the analyzed data context were 18 international cooperations characterized by 68 attributes such as factors of influence, typological qualities, and effects caused by the cooperations extracted from a multitude

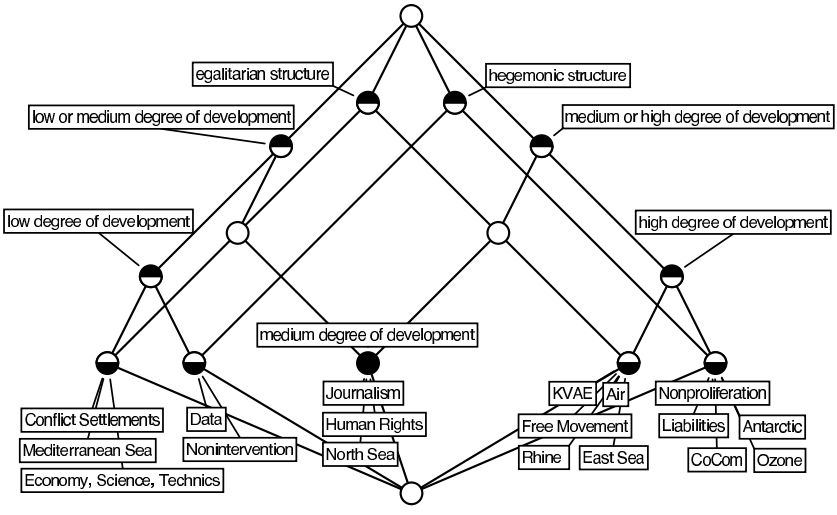


Fig. 7. Connection between the "degree of development" and the "power structure" of international cooperations

of case studies. The thorough examination of this data context using concept-analytic methods proved enormously valuable for the target of gaining insight into the requirements for the development and success of international cooperations [Ko89], [KV00].

Out of the fullness of the produced concept structures only one example shall be discussed here which refers to the following hypothesis widely mentioned in the literature: *"Strongly developed international cooperations can be mostly found under hegemonic structures."* In Fig. 7 the line diagram shows a concept lattice whose atoms name 18 power structures of international cooperations while the coatoms name the degree of development of those cooperations. The concept lattice makes visible that the cooperations with a high degree of development divide into 5 of egalitarian and 5 of hegemonic power structures and the cooperations with a low degree of development divide into 3 of egalitarian and 2 of hegemonic power structure, while all 3 cooperations with an intermediate degree of development have an egalitarian power structure. These findings seriously call the "hegemonic" hypothesis into question. During a presentation of TOSCANA on the CEBIT'93, it became obvious just how politically persuasive the hypothesis really is. W. Gerhardt, then minister of the Sciences and Humanities in the State of Hessen, expressed his astonishment about the insights gained by the project.

3.6 Analyzing

> *"Analyzing" means to theoretically examine facts with regard to their aim and objective.*

Data analysis, understood in this way, significantly influenced the development of Conceptual Knowledge Processing. In particular, the requirements resulting

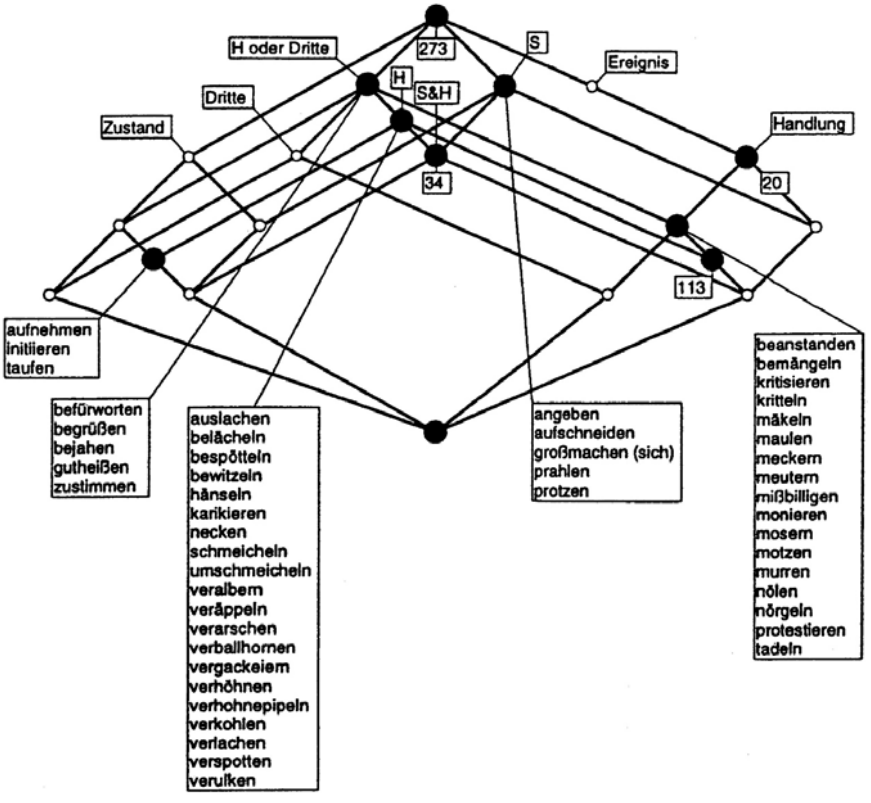


Fig. 8. The classification aspect "communication content" of a TOSCANA-system for speech act verbs

from this understanding gave rise to the TOSCANA-methodology for analyzing conceptual data. A convincing example for a purpose- and theory-guided examination design is the "TOSCANA-system for speech act verbs" which was established at the Institute for German Language (*Institut für deutsche Sprache*) in Mannheim in cooperation with the Darmstadt Research Group on Concept Analysis. The resulting system is designed to assist the *classification of speech acts*, a task formulated by the founder of the speech act theory, J. L. Austin, who provided the necessary distinctions of the verbs in question. The TOSCANA-methodology allows to bring together different, *language-logically structured aspects of classification* such as meaning, type of event, role allocation, time allocation etc. These can then be combined in one system which enables to examine the semantic structures of the speech act verbs with regard to an overall classification [GH00].

A small insight shall be given by the line diagram in Fig. 8 which illustrates the classification aspect "information content" (S = speaker, L = listener). In the

line diagram some of the attached numbers are made explicit by the corresponding list of the speech act verbs; for instance, the number 5 belonging to the circle with the label "S" is replaced by the five words "angeben", "aufschneiden", "gromachen (sich)", "prahlen", "protzen". These are the speech act words that refer solely to the speaker. The small, unblotted circle marked "event" is a so-called "lexical gap". "Event" exists for language theoretical reasons. In the German language however, there is no speech act verb which stands for an "event" and does not - at the same time - stands for an action. It would be interesting to know if there are languages that contain speech act verbs where the "event" circle is not a "lexical gap".

3.7 Making Aware

"Making aware" means to bring something to someone's consciousness.

Concept lattices again and again contribute to make facts, interrelations, and point of views more conscious. An impressive proof of this was given by N. Spangenberg in his habilitation thesis [Sp90]. In Spangenberg's research, conducted at the Center for Psychosomatic Medicine at the University of Giessen, concept lattices were used to evaluate *Repertory Grid Tests* to enable patients with eating disorders to become conscious of unsolved conflicts in their families; while doing so, Spangenberg cooperated with the mathematician K. E. Wolff [SW93].

In the course of the test, the patient names persons that are close to her and characterizes these persons in her own words (using adjectives). All such information is then represented in a formal context and its concept lattice. Finally, the concept lattice is presented to the patient by the therapist in the form of a line diagram which the patient and the therapist interpret together. How this may concretely look like shall be demonstrated by a small example. In the line diagram in Fig. 9 one can read that the set of objects of the underlying test context consists of the persons "I" (denoted by "myself" in Fig. 9), "ideal", "mother", "father", "sister", and "brother-in-law" ("ideal" is meant to mark the person that the "I" would like to be). The negative characteristics "superficial" and "easily offended", which the patient attributes to the mother and the father, but not to the I and the ideal, are conspicuous. The way the patient characterizes her immediate family is usually an important indication for the therapist. In this case this is done using the words "apprehensive", "difficult", and "withdrawn". To see such a negative evaluation clearly revealled (on the other hand positive traits are also reinforced) gives, supported by the therapist, the patient a chance to become aware of the conflicts with her family that may be reasons for this negative evaluation.

3.8 Deciding

"Deciding" shall mean to resolve a situation of uncertainty by an order.

As Conceptual Knowledge Processing aims at supporting people's rational thinking, judgment and action, it has led to the development of methods which are

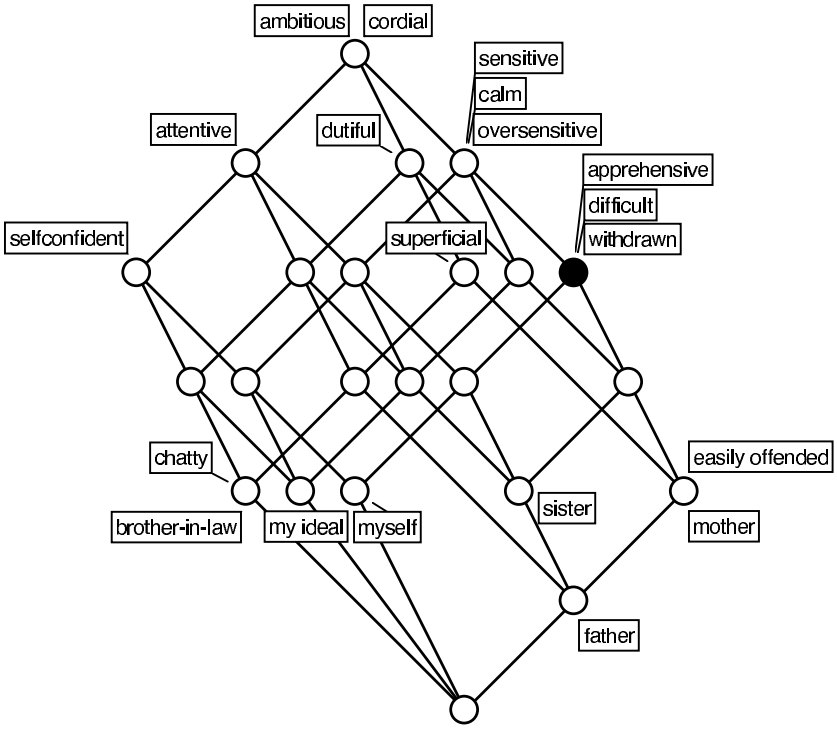


Fig. 9. Concept-analytical evaluation of a repertory grid test of an anorectic patient

particularly helpful for *decision making*. Already the concept lattices represented by line diagrams, by which the inherent conceptual relationships in data contexts become transparent, have multifariously proven as means of supporting decision making. How this can be performed in practice shall be discussed using data of the *National Water Research Institute* in *Burlington (Ontario)*: the objective was to determine which towns on the Canadian coast of Lake Ontario have locations suitable for swimming and tapping drinking water.

The concept lattice in Fig. 10 shows test values (concept-analytically: many-valued attributes) of 5 water samples of each of the 26 places (concept-analytically: objects) along Lake Ontario from the mouth of the Niagara River to the Cataraqui River/Kingston (see [SW92]). As the level of the pollution of the water rises, as the lines go down, the line diagram of the concept lattice shows that the places "STP Outfall", "Humber River", "Mouth of Credit River", "Cataraqui River", "Etobikoke Creek", and "Inside Bay" are extremely polluted ("Fecal Coliform 500" resp. "Coliphage 250"). If an "E. Coli" test value higher than 100 was considered harmful for swimmers, places "Bronte Creek", "Outfall area", and "Sunnyside Beach" would be banned for swimming. Making the decision - is swimming permitted or not - depend on the test values of 5 samples (criteria) is a case of *"multicriteria decision making"*. In Conceptual Knowledge Processing

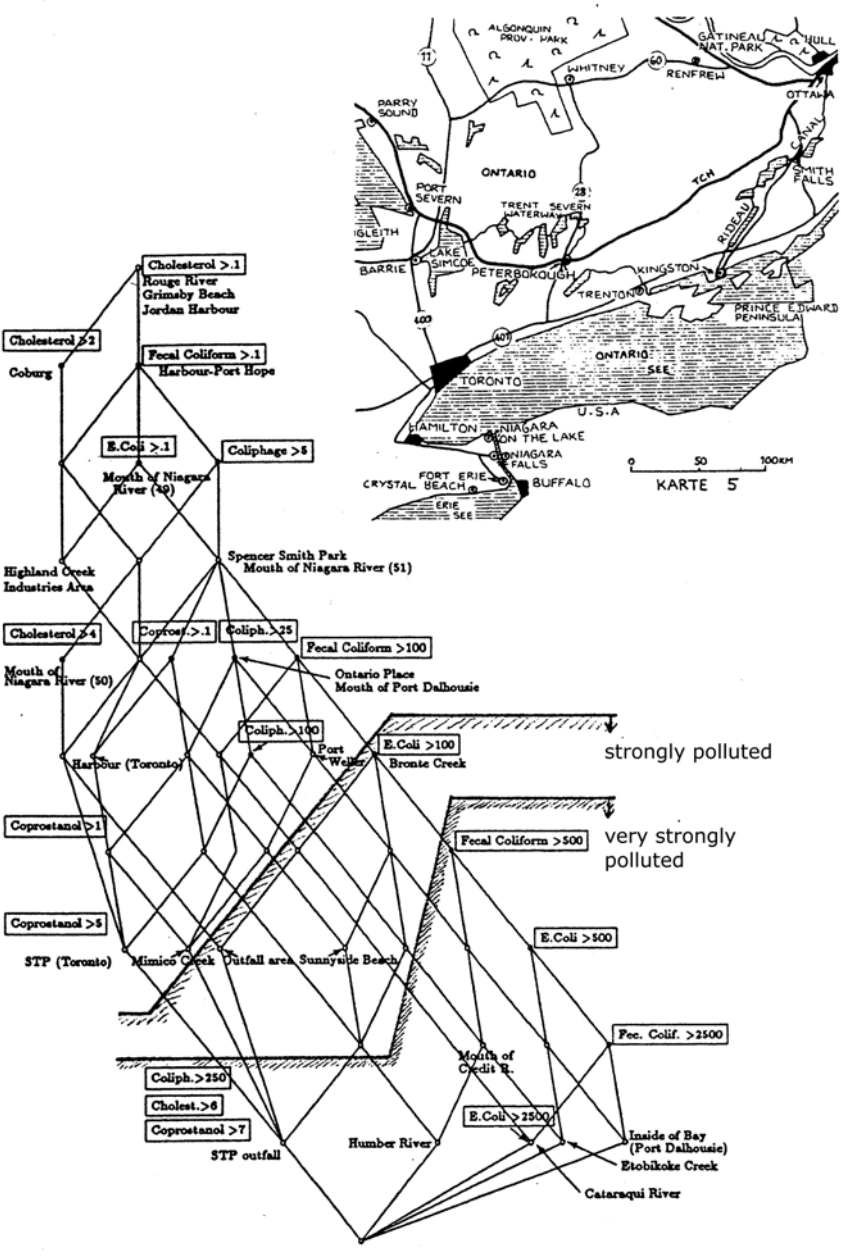


Fig. 10. Concept lattice about water pollution at the Canadian coast of Lake Ontario

however, the arithmetical mean of the test values is not used to make the decision, in the contrary the context of the test values is concept-analytically made transparent in a way that enables the people in question to come to a decision which is reasonable in its contents.

3.9 Improving

"Improving" has the meaning of enhancement in quality and value.

A requirement for successful improvement is that the facts and the interrelations concerned are well understood. This is supported by methods of Conceptual Knowledge Processing which shall be elucidated by an example *optimizing a production of electronic chips*. In the example discussed in [WS93], the outer quality of the chips produced was characterized by the parameters "final weight" z3, "breadth of a split" z2, and "weightiness" z1 := 100 (z2:z3). These parameters were examined as to their dependance on the four variables "temperature", "voltage", "catalyst", and "operator". To do so the 4 variables were set to all possible combinations of the three levels "low", "medium", and "high" which resulted in 81 test runs.

In Fig. 11 the results of the 81 test runs are depicted in a line diagram using three steps. The steps comprise the scales of quality shown individually in the middle of the diagram. The rating of the test values - from poor to good - starts on the right side (bottom), goes up and then on the left side down again (according to this the optimum of the values for z3 are lower than 190, for z2 they are higher than 30 and for z1 they are higher than 13). In order to optimize the setting of the parameters, one must first refer to the large rectangle on the bottom of the left side which shows the 16 test runs with the best values for z1. Of these test runs, 2 show the best values for z3 (and almost the best values for z2) as well as 4 times the best values for z2 of which only 1 value for z3 nearly belongs to the best. By doing so, 3 test runs have been identified, the settings of which come close to the best settings for the parameters that are possible. A more meticulous analysis of the test runs promises to possibly produce even better settings for the parameters.

3.10 Restructuring

"Restructuring" means to structurally design something new by certain measures.

Restructuring was adapted successfully in Software Engineering using methods of Conceptual Knowledge Processing [LS00]. The target of *software restructuring* is a transformation and reorganisation of the codes of programs in order to revive a system. In order to do so, basic principles of software engineering such as anticipation of changes and modularization must be implied taking criteria such as high cohesion and light coupling of modules into consideration.

To give an example, the *x-window-tool "xload"*, which consists of 724 lines of macro code, will be discussed. According to expert opinion, the program is

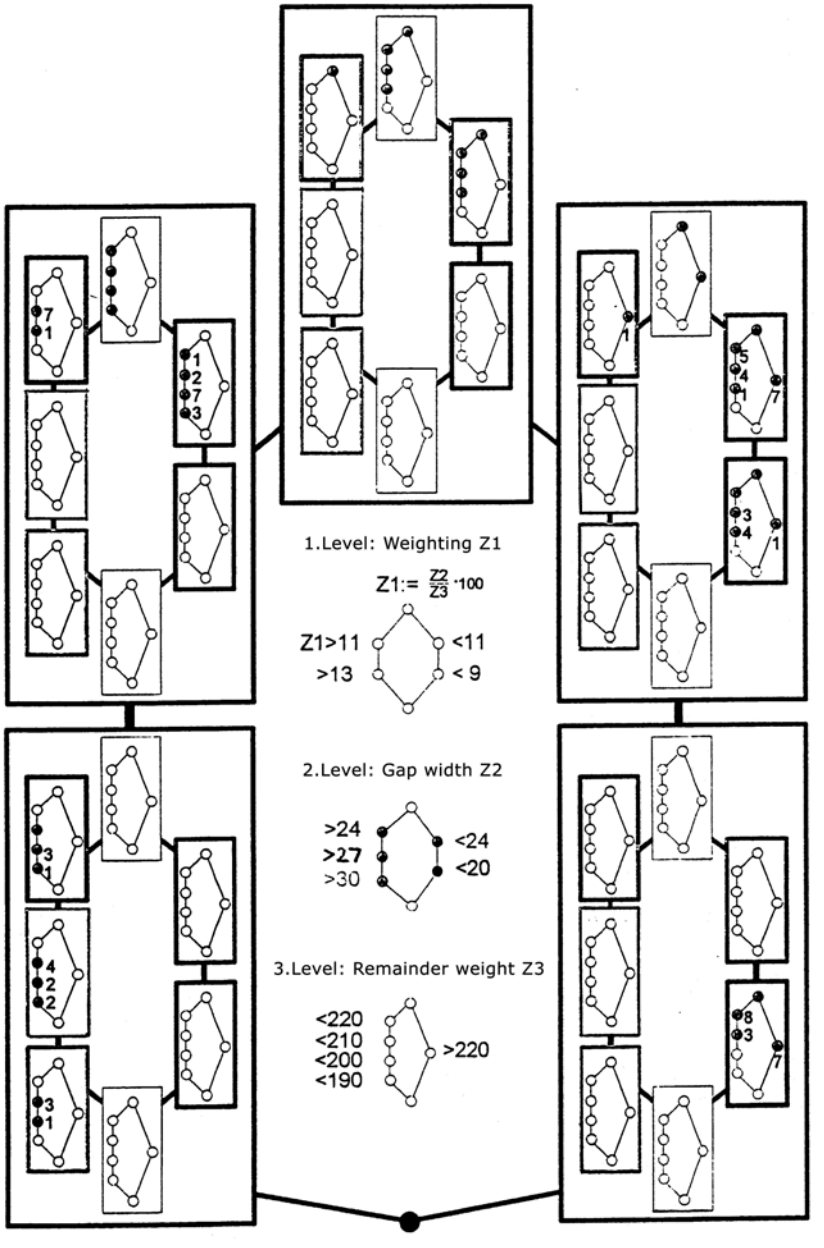


Fig. 11. A triply nested line diagram used for optimizing a chip production

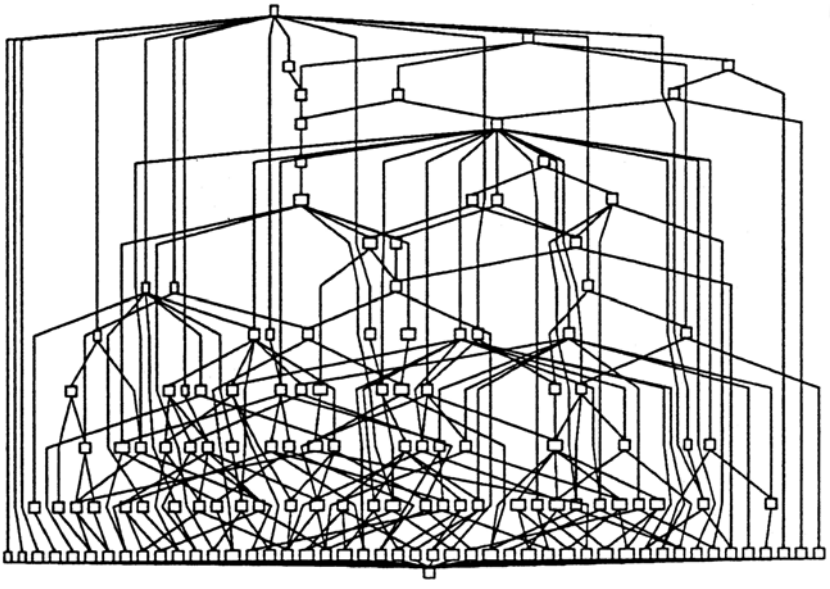


Fig. 12. Configuration lattice of the X-Window-Tools "xload"

not understandable by examination. Further, documentation does not exist for xload, changes and extensions are defected by a high probability of mistakes. The configuration lattice of "xload" with more than 100 elements, which is shown in Fig. 12 proves that this program is *configuration hacking* in its purest form. This concept lattice was used to restructure "xload"; in doing so, inexplicable configuration areas were modularizingly splitted into partial areas and governing terms of the program were simplified. After this restructuring (also called "amputation of irrelevant configurations") a macro code of 501 lines could be produced together with a clearly structured configutation lattice consisting of only 48 concepts.

3.11 Memorizing

> *"Memorizing" means to store something one has experienced or learned in order to use it later on.*

In Conceptual Knowledge Processing the so-called *conceptual data systems* (see [SVW93]), in which relational databases are combined with conceptual retrieval structures, are widely used to memorize information. How saving and reactivating of information can be achieved by conceptual data systems shall be explained using an musicological example of content analysis of text data [MW99]. As base for a doctoral thesis about the theme *"Simplicity. A Reconstruction of a Conceptual Landscape in the Aesthetics of Music of the 18th Century"*, a conceptual data system was elaborated and implemented as a TOSCANA-system. In order

to achieve this, 270 historical sources were searched with regard to a normed vocabulary consisting of more than 400 text features on the subject of "simplicity". These features were saved together with their classifications, and overriding categories were established for a large number of retrieval structures. This structuring of founded contents has to be understood as a *process of establishing a theory* which continually gains adherence by repeated checking of the text sources (aided by the examination of diagrams from the retrieval structures) [Ma00].

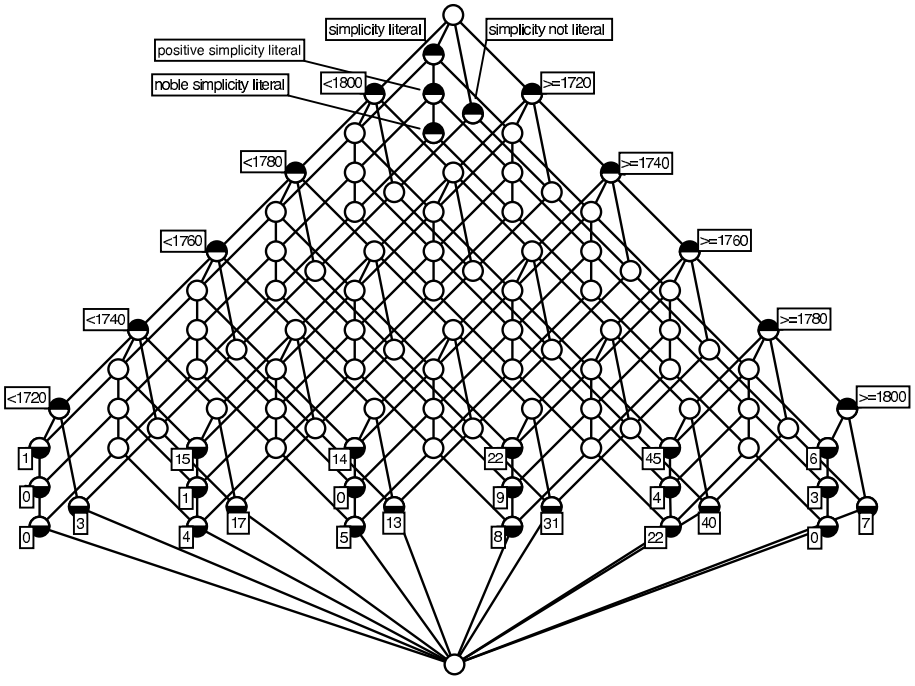


Fig. 13. Section of a "conceptual landscape" concerning the music esthetics of the 18th century

The line diagram in Fig. 13 stems from a *"conceptual landscape"* (cf. [Wi99]) elaborated in this way. The meet-irreducible elements are labelled by "< 1720", "< 1740", "< 1760", "< 1780", "< 1800", ">= 1720", ">= 1740", ">= 1780", ">= 1800", "simplicity literal", "simplicity not literal", "positive simplicity literal" and "noble-simplicity literal" as pictured in Fig. 13. The diagram discloses that the instances of the phrase "(noble-) simplicity" are most frequent from 1780 through 1800. "Simplicity" appears 45 times and "noble-simplicity" appears 22 times in the 111 text sources of this time span. The TOSCANA-system not only supports the recollection of the historic context that the terms were used, but also supports the process of categorizing and establishing a theory. This process is reflected in the retrieval structures that are developed.

3.12 Informing

"Informing" means to communicate knowledge about something to someone.

It has long been a matter of concern for conceptual knowledge processing to develop an extensive theory and effective methods for conceptual information and knowledge systems. In the last ten years it has become increasingly important to include relational contexts into the formal conceptual basis of such systems in a systematic manner. The developed *contextual logic* [Wi00a] serves as the theoretical basis for the extended theory which makes able formal descriptions of knowledge representations by concept graphs and information maps and allows formal derivations from data [EGSW00]. Its methods can be applied to information systems such as "information about air connections", "orientation in environments for learning", "search for literature in libraries" and many others.

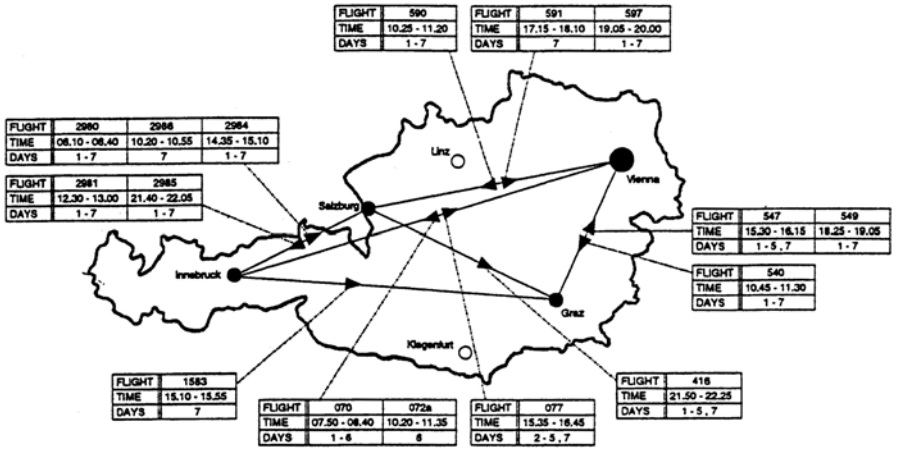


Fig. 14. Information map of flight connections in Austria

Fig. 14 shows an *information map for air connections* in Austria, in particular for the usecase of a passenger who lives in Vienna and wants to meet his business partners over a weekend in Graz, Innsbruck, and Salzburg. The information map shows possible flights for a departure from Vienna after 07:00 on a Saturday and for an arrival in Vienna before 20:00 on the following Sunday. It is easily inferred that the traveler has two main (but distinctly different) options [EGSW00]. The traveler will probably make his final decision for one flight or the other according to reasons that are not to be found in the map. The program for information maps for air connections used for Fig. 14 was developed by B. Groh who also programmed such an information map for the domestic flights in Australia.

4 Summary

Conceptual knowledge processing is obliged to a pragmatical understanding of knowledge according to which knowledge develops from an unlimited process of human thinking, arguing, and communicating, and lives on throughout this process. Theoretically, it is based upon *contextual logic* which aims for mutual combination of form and contents. This theoretical concept determines conceptual knowledge processing in practice with lasting effect. The significance of conceptual knowledge processing lies in its support for *general acts of thinking* such as exploring, searching, recognizing, identifying, investigating, analyzing, making aware, deciding, improving, restructuring, memorizing, informing, and others.

References

[BH05] Becker, P., Hereth Correia, J.: The ToscanaJ Suite for Implementing Conceptual Information Systems. In: FCA 2005. LNCS (LNAI), vol. 3626, pp. 324–348. Springer, Heidelberg (2005)

[De99] Devlin, K.: Infosense. Turning information into knowledge. Freeman, New York (1999)

[EGSW00] Eklund, P., Groh, B., Stumme, G., Wille, R.: A contextual-logic extension of TOSCANA. In: Ganter, B., Mineau, G.W. (eds.) ICCS 2000. LNCS (LNAI), vol. 1867, pp. 453–467. Springer, Heidelberg (2000)

[EKSW00] Eschenfelder, D., Kollewe, W., Skorsky, M., Wille, R.: Ein Erkundungssystem zum Baurecht: Methoden der Entwicklung eines TOSCANA-Systems. In: [SW00], pp. 254–272

[GH00] Großkopf, A., Harras, G.: Begriffliche Erkundung semantischer Strukturen von Sprechaktverben. In: [SW00], pp. 273–295

[GW99] Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Heidelberg (1999)

[He00] Hereth, J.: Formale Begriffsanalyse im Data Warehousing. Diplomarbeit. FB Mathematik, TU Darmstadt (2000)

[HSW00] Hereth, J., Stumme, G., Wille, R., Wille, U.: Conceptual knowledge discovery and data analysis. In: Ganter, B., Mineau, G.W. (eds.) ICCS 2000. LNCS (LNAI), vol. 1867, pp. 421–437. Springer, Heidelberg (2000)

[Ko89] Kohler-Koch, B.: Zur Empirie und Theorie internationaler Regime. In: Kohler-Koch, B. (Hrsg.) Regime in den internationalen Beziehungen, pp. 15–85. Nomos, Baden-Baden (1989)

[KV00] Kohler-Koch, B., Vogt, F.: Normen- und regelgeleitete internationale Kooperationen - Formale Begriffsanalyse in der Politikwissenschaft. In: [SW00], pp. 325–340

[KSVW94] Kollewe, W., Skorsky, M., Vogt, F., Wille, R.: TOSCANA - ein Werkzeug zur begrifflichen Analyse und Erkundung von Daten. In: [WZ94], pp. 267–288

[LS00] Lindig, C., Snelting, G.: Formale Begriffsanalyse im Software Engineering. In: [SW00], pp. 151–175

[Ma00] Mackensen, K.: Simplizität. Genese und Wandel einer musikästhetischen Kategorie des 18. Jahrhunderts. Bärenreiter, Kassel (2000)

[MW99] Mackensen, K., Wille, U.: Qualitative text analysis supported by conceptual data systems. Quality & Quantity 33, 135–156 (1999)

[PW99] Prediger, S., Wille, R.: The lattice of concept graphs of a relationally scaled context. In: Tepfenhart, W., Cyre, W. (eds.) ICCS 1999. LNCS (LNAI), vol. 1640, pp. 401–414. Springer, Heidelberg (1999)

[PRR99] Probst, G., Raub, S., Romhardt, K.: Wissen managen: wie Unternehmen ihre wertvollste Ressource optimal nutzen. 3. Aufl. Gabler, Wiesbaden (1999)

[RW00] Rock, T., Wille, R.: Ein TOSCANA-Erkundungssystem zur Literatursuche. In: [SW00], pp. 239–253

[SVW93] Scheich, P., Skorsky, M., Vogt, F., Wachter, C., Wille, R.: Conceptual data systems. In: Opitz, O., Lausen, B., Klar, R. (eds.) Information and Classification, pp. 72–84. Springer, Heidelberg (1993)

[So84] Sowa, J.F.: Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, Reading (1984)

[Sp90] Spangenberg, N.: Familienkonflikte essgestörter Patientinnen. Eine empirische Untersuchung mit der Repertory Grid Technik. Habilitationsschrift, Universität Gießen (1990)

[SW93] Spangenberg, N., Wolff, K.E.: Datenreduktion durch die Formale Begriffsanalyse von Repertory Grids. In: Scheer, J.W., Catina, A. (Hrsg.) Einführung in die Repertory Grid Technik. Bd.2: Klinische Forschung und Praxis. Huber, Bern, pp. 38–54 (1993)

[SW92] Strahringer, S., Wille, R.: Towards a structure theory for ordinal data. In: Schader, M. (ed.) Analyzing and Modeling Data and Knowledge, pp. 129–139. Springer, Heidelberg (1992)

[SW00] Stumme, G., Wille, R. (Hrsg.): Begriffliche Wissensverarbeitung: Methoden und Anwendungen. Springer, Heidelberg (2000)

[Wi82] Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Rival, I. (ed.) Ordered Sets, pp. 445–470. Reidel, Dordrecht (1982) (reprinted in: LNAI 5548, Springer, Heidelberg 2009)

[Wi87] Wille, R.: Bedeutungen von Begriffsverbänden. In: Ganter, B., Wille, R., Wolff, K.E. (Hrsg.) Beiträge zur Begriffsanalyse, pp. 161–211. B.I.-Wissenschaftsverlag, Mannheim (1987)

[Wi94] Wille, R.: Plädoyer für eine philosophische Grundlegung der Begrifflichen Wissensverarbeitung. In: [WZ94], pp. 11–25

[Wi99] Wille, R.: Conceptual landscapes of knowledge: a pragmatic paradigm for knowledge processing. In: Gaul, W., Locarek-Junge, H. (eds.) Classification in the Information Age, pp. 344–356. Springer, Heidelberg (1999)

[Wi00a] Wille, R.: Contextual Logic summary. In: Stumme, G. (ed.) Working with Conceptual Structures. Contributions to ICCS 2000, Shaker, Aachen, pp. 265–276 (2000)

[Wi00b] Wille, R.: Begriffliche Wissensverarbeitung: Theorie und Praxis. Informatik Spektrum 23, 357–369 (2000); gekürzte Version In: Schmitz B. (Hrsg.): Thema Forschung: Information, Wissen, Kompetenz. Heft 2/2000, TU Darmstadt, pp. 128–140

[Wi08] Wille, R.: Formal Concept Analysis and Contextual Logic. In: Hitzler, P., Schärfe, H. (eds.) Conceptual structures in practice. Chapman and Hall/CRS Press (to appear)

[WZ94] Wille, R., Zickwolff, M. (Hrsg.): Begriffliche Wissensverarbeitung: Grundfragen und Aufgaben. B.I.-Wissenschaftsverlag, Mannheim (1994)

[WS93] Wolff, K.E., Stellwagen, M.: Conceptual optimization in the production of chips. In: Janssen, J., Skiadas, C.H. (eds.) Applied Stochastic Models and Data Analysis, vol. II, pp. 1054–1064. World Scientific Publ. Comp., Singapore (1993)

Non-symmetric Indiscernibility

Bernhard Ganter

Technische Universität Dresden

Abstract. Rough Sets were introduced to express approximations based on an indiscernibility equivalence relation (Pawlak [4,5]). They have a natural lattice structure, which can nicely be described and widely generalised in the language of Formal Concept Analysis [2]. One instance of such a generalisation seems to be particularly promising: That of an indiscernibility preorder. The mathematical theory is almost the same as in the case of an equivalence relation, and some of the applications can be carried over. However, using preorders as indiscenibility relations needs getting used to, since such relations are not necessarily symmetric. We give an introduction and clarify the role of isolated and singleton elements.

1 Indiscernibility

The notion of indiscernibility is fundamental for the theory of Rough Sets, where (in the most basic case[1]) it is expressed by an equivalence relation on a set U, called the *universe*. The elements of the universe usually correspond to data base objects and therefore have *attributes*, and two objects are called indiscernible if they have exactly the same values for all attributes under consideration.

In the language of Formal Concept Analysiswe express this setting as follows: For a given formal context (U, M, I), define the **indiscernibility equivalence relation** $\sim$ on U by

$$g \sim h :\iff g' = h'.$$

This indeed yields an equivalence. The equivalence classes of this relation and their unions are the **definable sets**. Any subset $A \subseteq U$ of the universe contains a largest definable set $\underline{R}(A)$, and there is a smallest definable set $\overline{R}(A)$ containing it. These sets are called the **lower** and the **upper approximation** of A. Formally,

$$\underline{R}(A) := \{u \in U \mid \forall_{x \sim u}\ x \in A\}$$
$$\overline{R}(A) := \{u \in U \mid \exists_{x \sim u}\ x \in A\}.$$

The pair $(\underline{R}(A), \overline{R}(A))$ is the **rough set approximation** of A. Note that both $\underline{R}(A)$ and $\overline{R}(A)$ are definable sets, and that every definable set A satisfies $\underline{R}(A) = A = \overline{R}(A)$.

We suggest a generalised notion of indiscernibility by dropping the condition of symmetry. We claim that this is rather natural in absence of the *negation* of attributes. Given a formal context (U, M, I) and objects $g, h \in U$, let us say that

g is $\geq$*–discernible* from h iff g has an attribute that h does not have.

[1] Variations have extensively been studied, see [8].

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 26–34, 2011.

So g is $\geq$–discernible from h iff $g' \setminus h' \neq \emptyset$. Negating, we find that g is $\geq$–**indiscernible** from h iff $g' \subseteq h'$. This obviously defines a preorder. The equivalence relation obtained by intersecting this preorder with its dual is of course the original indiscernibility equivalence relation.

Calling a non-symmetric relation an indiscernibility relation is somewhat counterintuitive and may lead to confusion. Fortunately, this preorder is also known under a different name: It is the dual of the **object preorder** of the formal context (U, M, I), defined by

$$g \geq h :\iff g' \subseteq h' \iff \gamma g \geq \gamma h.$$

We therefore use the symbol $\geq$ for the indiscernibility preorder. This view also gives an alternative interpretation: g is indiscernible from h if h is a *specialisation* of g in the sense that h has all the attributes describing g. Then g is indiscernible from all its specialisations, but discernible from its proper generalisations. The preorder is therefore also called **specialisation preorder**.

The approximation operators can easily be generalised:

$$\underline{R}(A) := \{u \in U \mid \forall_{x \leq u}\, x \in A\}$$
$$\overline{R}(A) := \{u \in U \mid \exists_{x \geq u}\, x \in A\}.$$

So u belongs to the lower approximation of A iff every object that u is indiscernible from belongs to A. And u is in the upper approximation of A iff some element from A is indiscernible from u. In other words: In order to be in the lower approximation, the element and all its specialisations must belong to A. The upper approximation consists of all objects that are specialisations of some element of A.

Both the lower and the upper approximation contain with each element all elements it is indiscernible from, i.e., all its specialisations. In other words, each approximation is a *preorder ideal* in the object preorder. In the non-symmetric case these replace the definable sets.

We illustrate the definitions of the approximation operators by a small example. Let U be the set of all three-letter words made from letters a, b, and $*$, where $*$ is a wildcard replacing either a or b. Then U has 27 elements:

$$\begin{aligned} U = \{\ & aaa, aab, aa*, aba, abb, ab*, a*a, a*b, a**, \\ & baa, bab, ba*, bba, bbb, bb*, b*a, b*b, b**, \\ & *aa, *ab, *a*, *ba, *bb, *b*, **a, **b, *** \ \}. \end{aligned}$$

Structure this object set by the six attributes

"first letter is a", "first letter is b", "second letter is a", etc ...
(for a $*$, none of the attributes is given).

Now let

$$A := \{a*b, baa, bab, ba*, bba, bb*\}.$$

We find that

$$\underline{R}(A) = \{baa, bab, ba*, bba\}$$

and

$$\overline{R}(A) = \{aab, abb, a*b, baa, bab, ba*, bba, bbb, bb*\}.$$

2 Functional Dependencies and Indiscernibility

We have mentioned in Section 1 that indiscernibility equivalence relations often are derived from relational data bases. In the language of Formal Concept Analysissuch a data base corresponds to a **many-valued context** (G, M, W, I) with object set G, attribute set M, a set W of **attribute values** and a ternary relation $I \subseteq G \times M \times W$, such that $(g, m, w_1) \in I$ and $(g, m, w_2) \in I$ together imply $w_1 = w_2$. We read $(g, m, w) \in I$ as "the value of attribute m for object g is w", and abbreviate this occasionally as $m(g) = w$.

Two objects g and h of a many-valued context are *indiscernible* with respect to M if $m(g) = m(h)$ holds for all attributes $m \in M$. More generally g and h are indiscernible with respect to a subset $M_0 \subseteq M$ if $m(g) = m(h)$ holds for all $m \in M_0$. Obviously all these indiscernibility relations are equivalence relations on G.

A **functional dependency** $A \to B$ (where $A, B \subseteq M$) holds in a many-valued context if object pairs indiscernible with respect to A are always also indiscernible with respect to B. Formally, the condition is this: If $m(g) = m(h)$ for all $m \in A$, then also $m(g) = m(h)$ for all $m \in B$. It is somewhat self-suggesting how to connect functional dependencies with formal context implications: From a given many-valued context (G, M, W, I) construct a formal context $(G \times G, M, I_{\mathbb{N}})$ by

$$(g, h)\ I_{\mathbb{N}}\ m :\Longleftrightarrow\ m(g) = m(h).$$

Without problems it can be proved that the implications holding in this context are precisely the functional dependencies of (G, M, W, I).

It is worthwhile to describe the formal concepts of the context $(G \times G, M, I_{\mathbb{N}})$ obtained above. Each such formal concept (A, B) has as its extent A a subset of $G \times G$, i.e., a relation on G, and as its intent a subset $B \subseteq M$ of attributes. Indeed, (A, B) is a formal concept of $(G, M, I_{\mathbb{N}})$ precisely if A is the indiscernibility corresponding to B and at the same time B is the maximal attribute set inducing this indiscernibility relation. We call such a maximal set a **coreduct**, in analogy to a **reduct**, which in Rough Set Theory is a *minimal* set determining an indiscernibility relation.

3 Linguistic Variables

Formal Concept Analysis offers a more structural view of data than functional dependencies can express. Functional dependencies are based (only) on the equality or inequality of attribute values, but often there is more information available. For example, the values of an attribute may carry a (partial) **order**. This is the case both for *numerical* attribute values and for the values of a so-called **linguistic variable** (Zadeh [7], see also Wolff [6]). There is no unified definition

of linguistic variables, the term refers to quantities with values from language, like "large" or "cold". Formal Concept Analysisoffers a natural formalisation of linguistic variables: Many-valued attributes. There is no restriction on what the values of a many-valued attribute should be. Words, or even sentences from language are permitted as well as numerical values.

The set of all values of a many-valued attribute may be structured. Some attribute values may imply others, some may be mutually exclusive. For example "very cold" is usually meant to be stronger than "cold", and to be exclusive to "hot". Such value constraints can be expressed in Formal Concept Analysisusing the method of **conceptual scaling**. For our purposes here it suffices however to assume that the attribute values are **preordered**, i.e., that there is a reflexive and transitive **subsumption** relation $\sqsubseteq$ on the values. $a \sqsubseteq b$ expresses that every object having attribute value a also (implicitly) has the value b. The subsumption order is actually the object order, as introduced above, of the conceptual scale corresponding to the linguistic variable.

The presence of such a subsumption relation suggests a non-symmetric interpretation of indiscernibility, as in the following example:

Consider a data set on persons with a linguistic variable for family relationships, the values of which include "mother" and "parent", and assume that "parent" subsumes "mother". Then every attribute-definable subset containing all parents automatically contains all mothers, but not vice versa. Therefore the attribute value "'parent"' cannot discern from "'mother"', but "'mother"' discerns from "'parent"'.

If one follows this view, then the natural new definition of the indiscernibility relation for a many-valued context

$$(G, M, (W, \sqsubseteq), I)$$

with ordered values is to call an object h indiscernible from an object g iff

$$m(g) \sqsubseteq m(h) \quad \text{for all } m \in M.$$

What we obtain is exactly the indiscernibility preorder of the derived formal context, as defined in Section 1.

The data base construction given in the previous section can easily be modified. From the given many-valued context $(G, M, (W, \sqsubseteq), I)$ construct the context

$$(G \times G, M, I_{\mathbb{O}})$$

by

$$(g, h)\ I_{\mathbb{O}}\ m :\Longleftrightarrow\ m(g) \sqsubseteq m(h).$$

The concept extents of this formal context are precisely the dual indiscernibility preorders with respect to their intents. The intents are precisely the coreducts for this notion of indiscernibility. The implications of $(G \times G, M, I_{\mathbb{O}})$ are the same as the **ordinal dependencies** of $(G, M, (W, \sqsubseteq), I)$, defined as follows: An attribute set $B \subseteq M$ is ordinally dependent on an attribute set $A \subseteq M$ iff whenever $m(g) \sqsubseteq m(h)$ holds for all m in A then it also holds for all $m \in B$. Note that ordinal dependency implies functional dependency. Therefore every intent of $(G \times G, M, I_{\mathbb{N}})$ is also an intent of $(G \times G, M, I_{\mathbb{O}})$.

4 Decision Making

Given a formal context (U, M, I) and a subset $A \subseteq U$, the task of decision making is to describe A in terms of attribute combinations. Usually, A is not a concept extent.

One builds decision rules, resembling implications, by selecting subsets $H \subseteq M$ of attributes in such a way that $H' \subseteq A$. In the theory of JSM-reasoning, such subsets are called **hypotheses** for A. Usually, A is described using several hypotheses H_1, H_2, ..., attempting to solve

$$A = H_1' \cup H_2' \cup \ldots$$

as well as possible.

Not every subset admits such a description. In fact, the sets A that can be described in the above-mentioned form

$$A = H_1' \cup H_2' \cup \ldots$$

are precisely the *definable sets* defined earlier, i.e., the preorder ideals of the object preorder. Indeed, every extent H' is a preorder ideal, and if A is a preorder ideal, then

$$A = \bigcup_{g \in A} (g')'.$$

Rough sets for non-symmetric indiscernibility relations therefore use the same definable sets for their approximations as JSM-theory can describe using (disjunctions of) hypotheses. We summarise our findings in the following theorem.

Theorem 1 (characterising the definable sets). *Let (U, M, I) be a formal context, let $\leq$ be its object preorder and let the lower and upper approximation operator be defined as above. Then the following families of sets are identical:*

1. *The images of the lower approximation operator,*
2. *the images of the upper approximation operator,*
3. *the unions of extents of (U, M, I),*
4. *the preorder ideals of the object preorder $\leq$,*
5. *the extents of $(U, U, \not\geq)$.*

These sets are called the definable object sets of (U, M, I). The union and the intersection of definable sets is definable.

Together with S. Kuznetsov we have worked out some computational aspects of hypothesis generation in terms of non-symmetric indiscernibility, see [3]

5 The Lattice of Rough Set Approximations

The definable sets, ordered by set inclusion, form a distributive complete lattice. The lattice operations are set union and intersection. The pairs (A, B) of definable sets are the elements of the direct square of this lattice and thereby form

a lattice as well. The rough set approximations $(\underline{R}(A), \overline{R}(A))$ are contained in this lattice, but do not necessarily form a sublattice, because the upper approximation of an intersection is not necessarily the intersection of the individual upper approximations. Instead, the rough set approximations *generate* a complete sublattice of the lattice of pairs of definable sets. We can say a little more: Since

$$\underline{R}(A) \subseteq \overline{R}(A)$$

holds for all sets $A \subseteq U$, the complete sublattice generated by such pairs must be contained in the order relation of the lattice of definable sets.

We have given elsewhere [2] a characterisation of this sublattice. It is the concept lattice of the formal context in Figure 1.

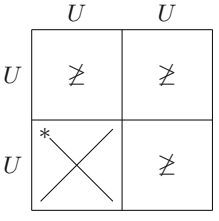


Fig. 1. The formal context for the lattice of rough set approximations in the case of an indiscernibility preorder. The relation in the lower left quadrant is $U \times U$, except for those pairs (u, u), where u is an isolated point, i.e., both minimal and maximal in the preorder.

The context construction given in [2] is more general. It is based on the fact that subdirect products of complete lattices can be described by so-called P-fusions of formal contexts.

That isolated points need special treatment is familiar already from the theory of indiscernibility equivalence relations, where they correspond to singleton equivalence classes. For preorders we even have to distinguish two cases:

Isolated points, that is, objects incomparable to all other objects. If u is isolated, then

$$u \in \underline{R}(A) \iff u \in A \iff u \in \overline{R}(A).$$

Moreover,

$$(\underline{R}(A \cup \{u\}), \overline{R}(A \cup \{u\})) = (\underline{R}(A) \cup \{u\}, \overline{R}(A) \cup \{u\}).$$

As a consequence, the lattice of rough set approximations splits into a direct product of the two-element lattice and the lattice of rough set approximations not containing u.

Singleton equivalence classes, consisting of those elements u which do not have a mutually indiscernible partner, i.e., an element $v \neq u$ such that $u \geq v$

and $v \geq u$. Let u be such an element, and assume that $\downarrow u \neq \{u\}$. Then both $\downarrow u \setminus \{u\}$ and $\downarrow u$ are definable sets, and they differ only by u. But then $(\downarrow u \setminus \{u\}, \downarrow u)$ cannot be of the form

$$(\downarrow u \setminus \{u\}, \downarrow u) = (\underline{R}(A), \overline{R}(A))$$

for some $A \subseteq U$. Note that
- for $A := \downarrow u \setminus \{u\}$ we get $(\underline{R}(A), \overline{R}(A)) = (\downarrow u \setminus \{u\}, \downarrow u \setminus \{u\})$, and
- for $A := \{u\}$ we get $(\underline{R}(A), \overline{R}(A)) = (\emptyset, \downarrow u)$, but

$$(\downarrow u \setminus \{u\}, \downarrow u \setminus \{u\}) \vee (\emptyset, \downarrow u) = (\downarrow u \setminus \{u\}, \downarrow u).$$

Some remarks must be made about generating the lattice of rough set approximations. It was said above that the pairs $(\underline{R}(A), \overline{R}(A))$ do not form, but merely generate a sublattice of the lattice of pairs of definable sets. This is irritating in two different ways: Firstly, we may discover that the set of such pairs, with the induced ordering, is indeed lattice ordered, but not a sublattice. Secondly, it may be asked if the properly generated elements have a meaning. Why should we consider pairs that do not actually approximate anything?

It turns out that the properly generated elements are exceptional cases, caused by singleton equivalence classes. Simply doubling such points removes the problem, as will be shown below. Before going into more detail, we demonstrate the effect by continuing a small example from [2]. Figure 2 shows an ordered set, representing the indiscernibility (pre-)order of our example. Next to it there is the lattice of its order ideals. These are, according to Theorem 1, precisely the definable sets.

The ordered set has 4 elements and consequently 16 subsets A, which indeed have mutually distinct rough set approximations $(\underline{R}(A), \overline{R}(A))$. They correspond to the shaded elements in Figure 3. We know from [2] that the lattice generated by the rough set approximations is isomorphic to the order of the lattice in Figure 2 (since there are no isolated elements). Indeed, there are 31 comparabilities in the lattice of definable sets (Figure 2) and consequently 31 elements in Figure 3. Approximately half of these elements are actually approximations. For example, $(\emptyset, \{f, m\})$ and $(\{f, m\}, \{f, m, p, w\})$ are generated, but are not rough set approximations.

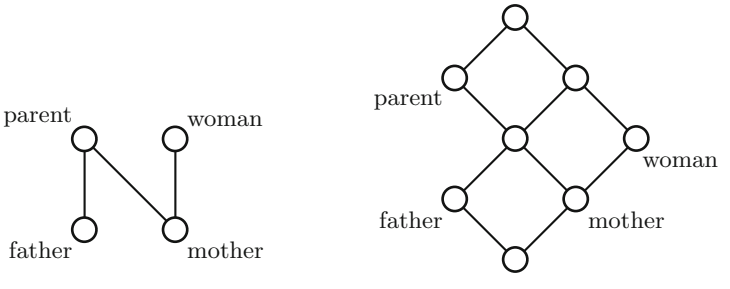


Fig. 2. An ordered set, and its lattice of order ideals (with object labels only)

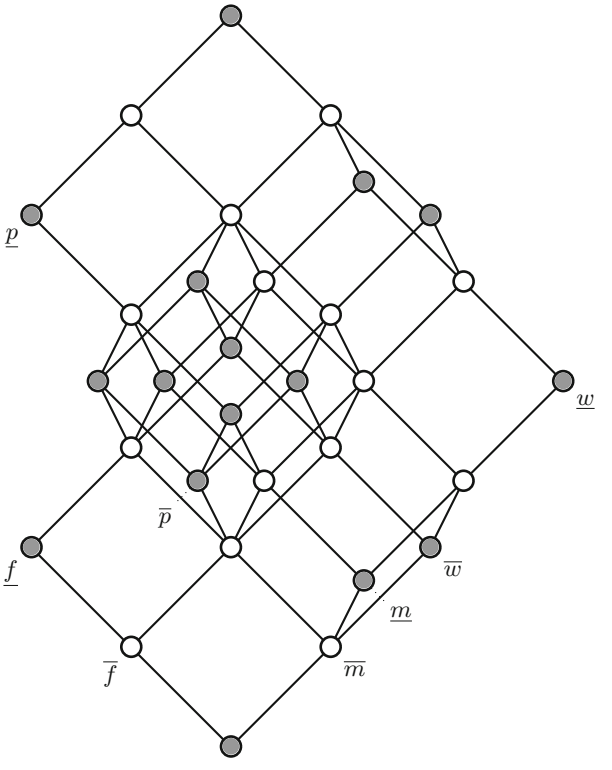


Fig. 3. The lattice of rough set approximations. Each concept extent E corresponds to a pair consisting of a lower and of an upper approximation. The lower approximation is formed by the underlined elements, the upper by the overlined elements in E. The shaded elements correspond to the pairs $(\underline{R}(A), \overline{R}(A))$, $A \subseteq \{f, m, p, w\}$.

Doubling all elements results in the same lattice structure. But now all lattice elements are rough sets. Figure 4 shows how the two above-mentioned elements are obtained.

Theorem 2. *An indiscernibility preorder has no singleton equivalence classes if and only if each pair in the sublattice generated by rough set approximations is itself a rough set approximation.*

Proof. Suppose there are no singleton equivalence classes. In particular then the preorder has no isolated points. The generated sublattice therefore consist of all pairs (L, U) of definable sets with $L \subseteq U$. Now let $A := L \cup R$, where R is a system of representatives of the equivalence classes of U (containing exactly one element from each equivalence class in U). Obviously, L is the largest union of equivalence classes contained in A, and U is the smallest union of equivalence classes containing A. But both U and L are definable sets. Therefore

$$(L, U) = (\underline{R}(A), \overline{R}(A)).$$

Fig. 4. The indiscernibility preorder from Figure 2, with doubled elements. Doubled elements have the same label and are mutually indiscernible. Choosing for A the set of marked elements yields $(\underline{R}(A), \overline{R}(A)) = (\emptyset, \{f, m\})$ for the left diagram and $(\underline{R}(A), \overline{R}(A)) = (\{f, m\}, \{f, m, p, w\})$ for the right one.

The converse was already discussed above: If $\{u\}$ is a singleton equivalence class, then $(\downarrow u \setminus \{u\}, \downarrow u)$ is an ordered pair of definable sets, but not of the form $(\underline{R}(A), \overline{R}(A))$ for any $A \subseteq U$. □

6 Conclusion

Non-symmetric indiscernibility relations, more precisely indiscernibility preorders, arise naturally from hierarchically structured data. Indiscernibility equivalence relations are obtained from scaling many-valued data *nominally*, while the general method of conceptual scaling leads to indiscernibility preorders. *Definable sets* turn out to be the preorder ideals. They form a distributive lattice, as in the case of an equivalence relation. The role of isolated points and singleton equivalence classes is more complicated, but can be handled.

References

1. Baixeries, J.: A Formal Concept Analysis Framework To Model Functional Dependencies. In: Proceedings of Mathematical Methods for Learning (2004)
2. Ganter, B.: Lattices of Rough Set Abstractions as *P*-Products. In: Medina, R., Obiedkov, S. (eds.) ICFCA 2008. LNCS (LNAI), vol. 4933, pp. 199–216. Springer, Heidelberg (2008)
3. Ganter, B., Kuznetsov, S.: Scale Coarsening as Feature Selection. In: Medina, R., Obiedkov, S. (eds.) ICFCA 2008. LNCS (LNAI), vol. 4933, pp. 217–228. Springer, Heidelberg (2008)
4. Pawlak, Z.: Rough Sets. International Journal of Computer and Information Sciences 11, 341–356 (1982)
5. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer Academic Publishing, Dordrecht (1991)
6. Wolff, K.E.: Concepts in Fuzzy Scaling Theory: Order and Granularity. Fuzzy Sets and Systems 132(1) (2002)
7. Zadeh, L.: The concept of a linguistic variable and its applications to approximate reasoning. Parts I–III, Information Sciences 8 (1975), 9 (1976)
8. See for an extensive bibliography, `http://roughsets.home.pl/`

Computing Graph-Based Lattices from Smallest Projections

Sergei O. Kuznetsov

State University Higher School of Economics (SU HSE),
Pokrovskii bd. 11, Moscow 109028, Russia
skuznetsov@hse.ru

Abstract. From the mathematical perspective, lattices of closed descriptions, which arise often in practical applications can be reduced to concept lattices by means of the Basic Theorem of Formal Concept Analysis (FCA). From the computational perspective, in many cases it is more advantageous to process closed descriptions and their lattices directly, without reducing them to concept lattices. Here a method for computing lattices with descriptions given by sets of graphs, starting with rough approximations is considered and compared to previous approaches.

1 Introduction

Recently, the problem of analyzing data given by labeled (hyper)graphs attracted much attention in various computer science communities due to its importance in many applications, from chemistry to text analysis [2, 9, 14, 15, 17, 20, 28–30]. Like in Data mining of 1990s the researchers came to the idea of a closed graph which can be very useful for defining association rules on graphs: As reported in [30], CloseGraph algorithm computes frequent graphs much faster than its forerunner gSpan [29], and WARMR [15], an ILP program.

As for FCA, a lattice on (closed) sets of labeled graphs, representing molecules, was proposed much earlier [18–21]. The lattice on graph sets is induced by an operation which takes a pair of graphs to the set of its maximal common subgraphs. This operation induces a meet (infimum) operation on sets of labeled (hyper)graphs: it is idempotent ($X \sqcap X = X$), commutative ($X \sqcap Y = Y \sqcap X$), and associative ($X \sqcap (Y \sqcap Z) = (X \sqcap Y) \sqcap Z$). These properties allow one to compute similarity of graph sets by means of algorithms for computing closed sets (see review [16]) well-known in Formal Concept Analysis [12].

In [11] we described a general framework, called pattern structures, which allows one to define similar lattices for sets of arbitrary partially-ordered descriptions and relate them to concept lattices using the Basic Theorem of FCA.

The main problem in practical implementation of these lattices is that of computational complexity, e.g., in case of graph sets even testing subgraph isomorphism is an NP-complete problem, the problem of computing common maximal subgraphs of two graphs being NP-hard.

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 35–47, 2011.

A theoretical means for approximate computation in semilattices, called projections of pattern structures, was proposed in [11] and the first computer implementation was described in [7].

The algorithm computing the lattice of graph sets described in [20] constructs the lattice (seen in the same perspective as a concept lattice) in a bottom-up way, starting from object "intents" (given by graphs). This algorithm is a simple modification of a standard FCA algorithm, however it does not realize the idea of constructing a sequence of projections, starting with the roughest one, and refining it until an appropriate level [11]. In standard FCA, this would be an algorithm which proceeds from attribute extents, however, one does not have attributes when working with graph sets.

In this paper we propose an algorithm which constructs the lattice of graph sets in a top-down way, starting from smallest subgraphs of the graphs in a dataset. We show the advantages of this algorithm over the previous, bottom-up algorithm.

The paper is organized as follows. In the second section we describe the general theoretical framework for computing similarity (meet) of graph sets together with a means for its approximate computations. In the third section we discuss a method for analyzing graph datasets based on the framework and discuss its drawbacks. In the fourth section we propose a new algorithm, discuss its complexity and advantages over the previous approach.

2 Pattern Structures on Sets of Graphs

In [18–20] a semilattice on sets of graphs with labeled vertices and edges was proposed and in [11] this semilattice was generalized to arbitrary pattern structures. As in the general case, the lattice on graph sets is based on a natural "containment" (i.e., in case of graphs, subgraph isomorphism) relation between graphs with labeled vertices and edges. Consider an ordered set P of connected graphs[1] with vertex and edge labels from the set $\mathcal{L}$ with partial order $\preceq$. Each labeled graph Γ from P is a quadruple of the form $((V, l), (E, b))$, where V is a set of vertices, E is a set of edges, $l\colon V \to \mathcal{L}$ is a function assigning labels to vertices, and $b\colon E \to \mathcal{L}$ is a function assigning labels to edges.

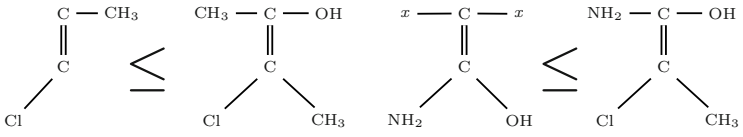
For two graphs $\Gamma_1 := ((V_1, l_1), (E_1, b_1))$ and $\Gamma_2 := ((V_2, l_2), (E_2, b_2))$ from P we say that Γ_1 **dominates** Γ_2 or $\Gamma_2 \leq \Gamma_1$ (or Γ_2 is a **subgraph** of Γ_1) if there exists an injection $\varphi\colon V_2 \to V_1$ such that it

- respects edges: $(v, w) \in E_2 \Rightarrow (\varphi(v), \varphi(w)) \in E_1$,
- fits under labels: $l_2(v) \preceq l_1(\varphi(v))$, $(v, w) \in E_2 \Rightarrow b_2(v, w) \preceq b_1(\varphi(v), \varphi(w))$.

Obviously, $(P, \leq)$ is a partially ordered set.

Example 1. Let $\mathcal{L} = \{\mathrm{C}, \mathrm{NH_2}, \mathrm{CH_3}, \mathrm{OH}, x\}$ then we have the following relations:

[1] Omitting the condition of connectedness, one obtains a (computationally harder) model that accounts for multiple occurrences of subgraphs.



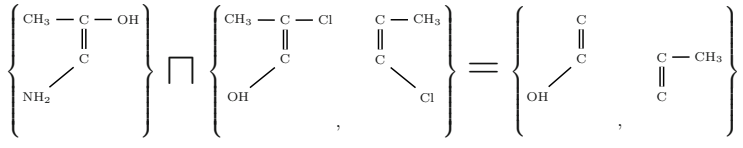
Now a *similarity operation* $\sqcap$ on graph sets can be defined as follows: For two graphs X and Y from P

$$\{X\} \sqcap \{Y\} := \{Z \mid Z \leq X, Y,\ \forall Z_* \leq X, Y\ \ Z_* \not\geq\ \ Z\},$$

i.e., $\{X\} \sqcap \{Y\}$ is the set of all maximal common subgraphs of graphs X and Y. Similarity of non-singleton sets of graphs $\{X_1, \ldots, X_k\}$ and $\{Y_1, \ldots, Y_m\}$ is defined as

$$\{X_1, \ldots, X_k\} \sqcap \{Y_1, \ldots, Y_m\} := \mathrm{MAX}_{\leq}(\cup_{i,j}(\{X_i\} \sqcap \{Y_j\})),$$

where $\mathrm{MAX}_{\leq}(X)$ returns maximal (w.r.t. $\leq$) elements of X. Here is an example of applying $\sqcap$:



The similarity operation $\sqcap$ on graph sets is commutative: $X \sqcap Y = Y \sqcap X$ and associative: $(X \sqcap Y) \sqcap Z = X \sqcap (Y \sqcap Z)$.

A set X of labeled graphs from P for which $\sqcap$ is idempotent, i.e., $X \sqcap X = X$ holds, is called a *pattern* of P. For patterns we have $\mathrm{MAX}_{\leq}(X) = X$. For example, for each graph $g \in P$ the set $\{g\}$ is a pattern. On the contrary, for $\Gamma_1, \Gamma_2 \in P$ such that $\Gamma_1 \leq \Gamma_2$ the set $\{\Gamma_1, \Gamma_2\}$ is not a pattern. Denote by D the set of all patterns of P, then $(D, \sqcap)$ is a semilattice with infimum (meet) operator $\sqcap$. The natural subsumption order on patterns is given by

$$c \sqsubseteq d :\Longleftrightarrow\ c \sqcap d = c.$$

Let E be a set of example names, and let $\delta : E \to D$ be a mapping, taking each example name to $\{g\}$ for some labeled graph $g \in P$ (thus, g is the "graph description" of example e). The triple $(E, (D, \sqcap), \delta)$ is a particular case of a *pattern structure* [11]. Another example of an operation $\sqcap$ may be the following semilattice on closed intervals from [19]: for $a, b, c, d \in R$, $[a, b] \sqcap [c, d] = [\max\{a, c\}, \min\{b, d\}]$ if $[a, b]$ and $[c, d]$ overlap, otherwise $[a, b] \sqcap [c, d] = \emptyset$. This semilattice, where numbers are values of the activation energy (computed for molecules by a standard procedure, e.g. see [32]) was used in predicting toxicity of alcohols and halogen-substituted hydrocarbons (see Section 4). The resulting similarity semilattice in this application is that on pairs, where the first element is a graph set and the second element is a numerical interval.

Derivation operators are defined as

$$A^\diamond := \sqcap_{e \in A}\ \delta(e) \qquad \text{for } A \subseteq E$$

and

$$d^\diamond := \{e \in E \mid d \sqsubseteq \delta(e)\} \qquad \text{for } d \in D.$$

For $a, b \in D$ the *pattern implication* $a \to b$ holds if $a^\diamond \subseteq b^\diamond$, and the *pattern association rule* $a \longrightarrow_{c,s} b$ with *confidence* c and *support* s holds if $\frac{|a^\diamond \cap b^\diamond|}{|G|} \geq s$ and $\frac{|a^\diamond \cap b^\diamond|}{|a^\diamond|} \geq c$. Implications are the exact association rules, i.e., association rules with confidence = 1. Operator $(\cdot)^{\diamond\diamond}$ is a closure operator on patterns, since it is

idempotent: $d^{\diamond\diamond\diamond\diamond} = d^{\diamond\diamond}$,
extensive: $d \sqsubseteq d^{\diamond\diamond}$,
monotone: $d^{\diamond\diamond} \sqsubseteq (d \sqcup c)^{\diamond\diamond}$.

For a set X the set $X^{\diamond\diamond}$ is called *closure* of X. A set of labeled graphs X is called *closed* if $X^{\diamond\diamond} = X$. This definition is related to the notion of a closed graph [30], which is important for computing association rules between graphs. Closed graphs are defined in [30] in terms of "counting inference" as follows.

Given a labeled graph dataset D, the support of a graph g or *support*(g) is the set (or the number) of graphs in D, in which g is a subgraph. A graph g is called *closed* if no supergraph f of g (i.e., a graph such that g is isomorphic to a subgraph of f) has the same support.

Note that the definitions distinguish between a closed graph g and the closed set $\{g\}$ consisting of one graph g. Closed sets of graphs form a *meet semilattice* w.r.t. the infimum or meet operator. A finite meet semilattice is completed to a lattice by introducing a unit (maximal) element. Closed graphs do not have this property, since in general, there can be no supremum and/or no infimum of two given closed graphs. Let a dataset described by a pattern structure $(E, (D, \sqcap), \delta)$ be given.

Proposition. The following two properties hold for a pattern structure $(E, (D, \sqcap), \delta)$:

1. For a closed graph g there is a closed set of graphs G such that $g \in G$.
2. For a closed set of graphs G and an arbitrary $g \in G$, graph g is closed.

Proof. 1. Consider the closed set of graphs $G = \{g\}^{\diamond\diamond}$. Since G consists of all maximal common subgraphs of graphs that have g as a subgraph, G contains as an element either g or a supergraph f of g. In the first case, property 1 holds. In the second case, we have that each graph in G that has g as a subgraph also has f as a subgraph, so f has the same support as g, which contradicts the fact that g is closed. Thus, $G = \{g\}^{\diamond\diamond}$ is a closed set of graphs satisfying property 1. 2. Consider a closed set of graphs G and $g \in G$. If g is not a closed graph, then there is a supergraph f of it with the same support as g has and hence, with the same support as G has. Since G is the set of all maximal common subgraphs of the graphs describing examples from the set $G^\diamond$ (i.e, its support), $f \in G$ should

hold. This contradicts the fact that $g \in G$, since a closed set of graphs cannot contain as elements a graph and a supergraph of it (otherwise, its closure does not coincide with itself). □

Therefore, one can use algorithms for computing closed sets of graphs, e.g., the algorithm in [20], to compute closed graphs. With this algorithm one can also compute all *frequent* closed sets of graphs, i.e., closed sets of graphs with support above a fixed *minsup* threshold (by introducing a minor variation of the condition that terminates computation branches).

Computing $\sqcap$ may require considerable computation resources: even testing $\sqsubseteq$ is NP-complete. To approximate graph sets we consider projection (kernel) operators [11], i.e. mappings of the form $\psi\colon D \to D$ that are

monotone: if $x \sqsubseteq y$, then $\psi(x) \sqsubseteq \psi(y)$,
contractive: $\psi(x) \sqsubseteq x$, and
idempotent: $\psi(\psi(x)) = \psi(x)$.

Any projection of the semilattice $(D, \sqcap)$ is $\sqcap$-preserving, i.e., for any $X, Y \in D$

$$\psi(X \sqcap Y) = \psi(X) \sqcap \psi(Y),$$

which helps us to relate learning results in projections to those with initial representation (see Section 3).

In our computer experiments in [22] we used several types of projections of sets of labeled graphs that are natural in chemical applications:

- *k-chain* projection: a set of graphs X is taken to the set of all chains with k vertices that are subgraphs of at least one graph of the set X;
- *k-vertex* projection: a set of graphs X is taken to the set of all subgraphs with k vertices that are subgraphs of at least one graph of the set X;
- *k-cycles* projection: a set of graphs X is taken to the set of all subgraphs consisting of k adjacent cycles of a minimal cyclic basis of at least one graph of the set X.

3 Analyzing Graph Datasets Using Lattice-Based Approaches

JSM-hypotheses were defined in [6] by means of a special logical language for standard object-attribute representation. These hypotheses were redefined as *JSM-* or *concept-based hypotheses* in [10, 11, 19, 20] in terms of Formal Concept Analysis (FCA). For graph sets hypotheses can be defined as follows. Suppose we have a set of positive examples E_+ and a set of negative examples E_- w.r.t. a target attribute.

A graph set $h \in D$ is a *positive hypothesis* iff

$$h^\diamond \cap E_- = \emptyset \text{ and } \exists A \subseteq E_+ : A^\diamond = h.$$

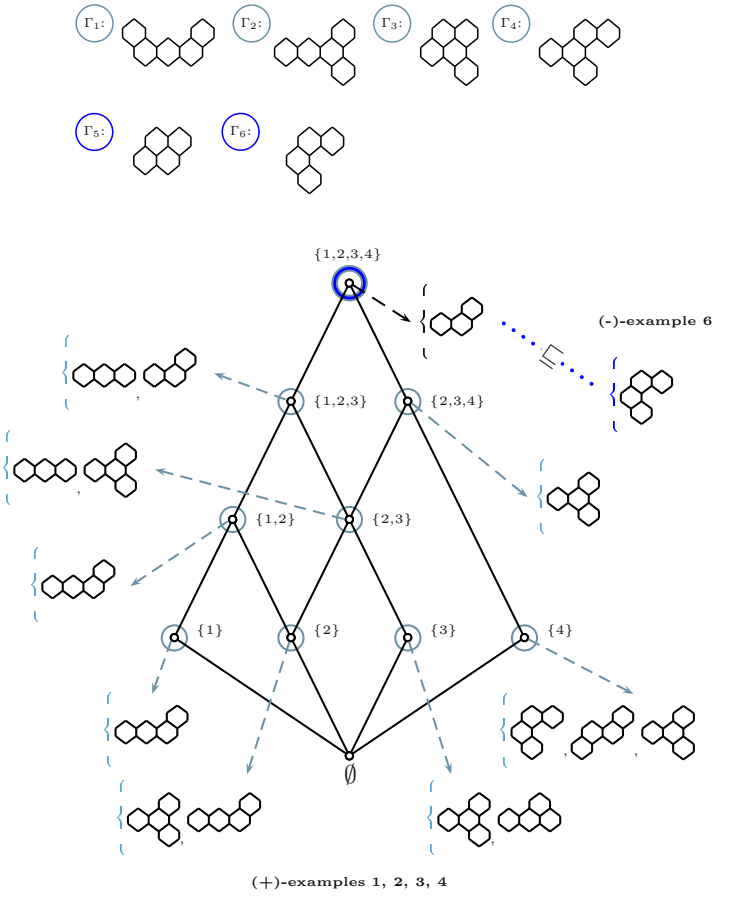


Fig. 1.

Informally, a positive hypothesis is a similarity of positive examples, which does not cover any negative example. A *negative hypothesis* is defined analogously, by interchanging + and −.

The meet-preserving property of projections implies that a hypothesis H_p in data under projection ψ corresponds to a hypothesis H in the initial representation for which the image under projection is equal to H_p, i.e., $\psi(H) = H_p$.

Hypotheses are used for classification of undetermined examples along the lines of [6] in the following way. If e is an undetermined example (example with the unknown target value), then a hypothesis h with $h \sqsubseteq \delta(e)$ is *for the positive classification* of g if h is a positive hypothesis and h is *for the negative classification* of e if h is a negative hypothesis.

An undetermined example e is *classified positively* if there is a hypothesis for its positive classification and no hypothesis for its negative classification. Example e is *classified negatively* in the opposite case. If there are hypotheses for

both positive and negative classification, then some other methods (e.g., based on standard statistical techniques) may be applied. Obviously, for classification purposes it suffices to use only hypotheses minimal w.r.t. subsumption $\sqsubseteq$.

Notwithstanding its simplicity, the model of learning and classification with concept-based hypotheses proved to be efficient in numerous computer experiments, including PTC competition [4, 25].

An algorithm for computing hypotheses on closed graph sets was described in [20]. In [22] the algorithm was realized by simulating $\sqcap$ operation with usual set-theoretic intersection $\cap$ in the following way. For each example e described by a labeled graph $\delta(e)$ first a set of all subgraphs of $\delta(e)$ is computed up to the projection level $k = N$. Each such subgraph is declared to be a binary attribute and example e is represented by the set $S(e)$ of binary attributes that correspond to subgraphs of $\delta(e)$. For two examples e_1 and e_2 the intersection $S(e_1) \cap S(e_2)$ is equivalent to finding the similarity $\psi(\delta(e_1)) \sqcap \psi(\delta(e_2))$.

Example 2. In Figure 1 consider JSM-hypotheses for the dataset with positive examples described by graphs Γ_1,..,Γ_4 and negative examples described by graphs Γ_5 and Γ_6. Here $\Gamma_1 \sqcap \Gamma_2 \sqcap \Gamma_3$ and $\Gamma_2 \sqcap \Gamma_3 \sqcap \Gamma_4$ are minimal positive hypotheses, whereas $\Gamma_1 \sqcap \Gamma_2 \sqcap \Gamma_3 \sqcap \Gamma_4$ is not a positive hypothesis.

Standard Weka procedures for C4.5, Naive Bayes and JRip were run [22] in QuDA environment [13, 26]. Computing concept-based hypotheses in QuDA is realized by means of algorithms for computing lattices of closed sets (or concept lattices), see review [16].

After that *reduction of attributes* [12] was performed: each column of the example/attribute binary table that is equal to the (component-wise) product (conjunction) of some other columns, was removed. Reduction guarantees [12] that thus reduced set of columns gives rise to the isomorphic lattice of closed sets of attributes and thus, to the same set of concept-based hypotheses as defined above.

The whole **PBRL** (project-binarize-reduce-learn) procedure proposed in [22] looks as follows:

1. For each example e and for k compute i-projections of $\delta(e)$ for $1 \leq i \leq k$. The subgraphs from these projections are declared to be attributes;
2. Compose example/attribute context;
3. For each learning method LM run LM, classify examples from test sets, compute cross validation;
4. Clarify and reduce the binary (example/attribute) context;
5. For reduced context and for each learning method LM run LM, classify examples from test sets, compute cross-validation.

Using this approach we analyzed several chemical datasets[2] in [22]. For each dataset we computed graph projections: mostly, k-vertex projections and k-cycles projections for the dataset with polycyclic aromatic hydrocarbons. Every subgraph of each graph in the projection (up to isomorphism) was declared to

[2] These datasets can be downloaded from `http://ilp05-viniti.narod.ru`

be a binary attribute, so each graph dataset was turned into a binary object-attribute table, which was then reduced. For each dataset we ran several learning methods realized within QuDA environment (JSM or concept-based hypotheses, induction of decision trees by C4.5, Naive Bayes, JRip) comparing the results with those based on same learning methods and a chemical (descriptor) attribute language, called FCSS [1], with predefined descriptors.

Although the results of comparison were in favor of graph representation, the major problem of this approach that we encountered in [22] was that of space complexity: although usually the reduced set of attributes was feasible, the initial set of attributes before reduction was too large. For databases we studied, we usually could not produce projections of size larger than 9 for all graphs in the database. By the definition of a projection, one cannot do it object-incrementally, but one can do it attribute-incrementally. For graph projections this means that one should start from smallest subgraphs of graphs from a dataset and proceed by increasing the graph size. Another point is that instead of considering k-projections of graphs, one can consider closures of these projections, since they are subgraphs of the same graphs in the dataset as the initial k-projections. Thus, we come up to the necessity of designing an algorithm that would construct the pattern lattice on graph sets in the top-down way: starting with smallest possible graphs.

4 A Top-Down Algorithm

In this section we propose a top-down algorithm for computing the set of all pattern concepts together with the covering relation on the set of concepts (i.e., graph of the diagram).

Assume that graph edges are unlabeled. This does not result in the loss of generality, since the case with labeled edges can be reduced to the one with labeled vertices. Let the set of vertex labels be $L = \{l_1, \ldots, l_n\}$, the vertex labeling function is denoted by $l(\cdot)$ and $\mathcal{G} = \{\Gamma_1, \ldots, \Gamma_m\}$ denotes a set of graphs. For $i \in [1, m]$, let $\Gamma_i = (V_i, E_i)$ and for $j \in [1, |V_i|]$ let v_i^j denote the jth vertex of Γ_i. Let also $k \in [1, n]$, and Λ be a special symbol.

if $k < n$ then $\mathbf{next}(\mathcal{G}, \Gamma_i, v_i^j, a_k) = (\mathcal{G}, \Gamma_i, v_i^j, a_{k+1})$ else
if $j < |V_i|$ then $\mathbf{next}(\mathcal{G}, \Gamma_i, v_i^j, a_k) = (\mathcal{G}, \Gamma_i, v_i^{j+1}, a_1)$ else
if $i < m$ then $\mathbf{next}(\mathcal{G}, \Gamma_i, v_i^j, a_k) = (\mathcal{G}, \Gamma_{i+1}, v_{i+1}^1, a_1)$
else $\mathbf{next}(\mathcal{G}, \Gamma_i, v_i^j, a_k) = \Lambda$.

The function "+" below realizes the idea of a "minimal extension" of a graph set. If we consider only connected subgraphs, the function $+(\mathcal{G}, \Gamma_i, v_i^j, a_k)$ is defined as follows: $+\Lambda = \Lambda$,

$$+(\mathcal{G}, \Gamma_i, v_i^j, a_k) := (\mathcal{G} \setminus \Gamma_i) \cup \Gamma_i^*,$$

where

$$\Gamma_i^* = (V_i^*, E_i^*), V_i^* = V_i \cup \{v\}, E_i^* = E_i \cup (v_i^j, v), v \notin V_i, l(v) = a_k.$$

If one considers not necessarily connected subgraphs, then the definition of **next** and "+" should be made independent of the "connecting" vertex v_i^j, e.g., for "+" operation as

$$+(\mathcal{G}, \Gamma_i, a_k) := (\mathcal{G} \setminus \Gamma_i) \cup \Gamma_i^*,$$

where

$$\Gamma_i^* = (V_i^*, E_i), V_i^* = V_i \cup \{v\}, v \notin V_i, l(v) = a_k.$$

Further on we consider only the case of connected subgraphs, the case with general subgraphs is treated similarly.

By definitions, $(+\mathbf{next}(\mathcal{G}, \Gamma_i, v_i^j, a_k))^\diamond$ is the set of objects with common description $(+\mathbf{next}(\mathcal{G}, \Gamma_i, v_i^j, a_k))$, i.e., a pattern extent. The pattern

$$(+\mathbf{next}(\mathcal{G}, \Gamma_i, v_i^j, a_k))^{\diamond\diamond}$$

is the closure of the pattern

$$(+\mathbf{next}(\mathcal{G}, \Gamma_i, v_i^j, a_k)).$$

We set by definition $\Lambda^{\diamond\diamond} = \Lambda^\diamond = \emptyset$.

If X stays for a pattern concept $(\mathcal{G}^\diamond, \mathcal{G})$ the function $\mathbf{covers}(X)$ computes concepts which X covers in the lattice order, i.e., taking the set

$$\bigcup_{i,j,k} (+(\mathbf{next}(\mathcal{G}, \Gamma_i, v_i^j, a_k)))^\diamond$$

it returns concepts that correspond to maximal extents of this set. This function can be efficiently realized (in $O(|X|^2 \cdot |G|)$ time), since extents are given as tuples.

There can be various efficient implementations of storing concepts together with the covering relation on them. For the sake of simplicity we consider here the one, consisting of the (lexicographically ordered) extent list, where from each extent there is a pointer to the corresponding intent and a pointer to the set of concepts covered by the pattern concept with this extent. So, the function **save**(X,covers(X)) takes a concept X, the corresponding set $\mathbf{covers}(X)$ and inserts extent of X in the list of extents, making pointers to intent of X and elements of $\mathbf{covers}(X)$ in the list of extents.

The top-down algorithm for graph lattices (TDAGL) traverses the diagram of the lattice in a standard depth-first way. The right margin shows the worst-case complexity of the algorithms steps

tdagl(X):
  mark X as **visited** $O(1)$
  process(X) $O((\alpha + \beta) \cdot \max_i |V_i| \cdot m \cdot n \cdot |G|)$
  for all $Y \in \mathbf{covers}(X)$ and Y not **visited** do
    tdagl(Y)

The procedure **process**(X) for $X = (\mathcal{G}^\diamond, \mathcal{G}^{\diamond\diamond})$ is described as follows:

process(X)
  for all i, j, k
    compute $(+\text{next}(\mathcal{G}, \Gamma_i, v_i^j, a_k))^{\diamond\diamond}$ $O(\alpha \cdot \max_i |V_i| \cdot m \cdot n \cdot |G|)$
    compute $(+\text{next}(\mathcal{G}, \Gamma_i, v_i^j, a_k))^{\diamond}$ $O(\beta \cdot \max_i |V_i| \cdot m \cdot n \cdot |G|)$
  save$(X,$ **covers**$(X))$ $O(|G|)$
  for all $Y \in$ **covers**(X) do
    if extent$(Y) \notin$ extentlist then **insert**(extent(Y), extentlist)
    $O(|G| \cdot |\textbf{covers}(X)|)$.

4.1 Top-Down and Bottom-Up Algorithms

The total worst-case complexity of TDAGL, which constructs the set of all pattern concepts together with the covering relation on them is

$$O((\alpha + \beta) \cdot | \prec | \cdot \max_i |V_i| \cdot m \cdot n \cdot |G|),$$

where $| \prec |$ is the size of the covering relation $\prec$ on pattern concepts (i.e., the number of edges in the lattice diagram). This complexity is similar to that of the bottom-up algorithm.

If we compare computing graph-based hypotheses with a bottom-up algorithm from [20] and TDAGL, we see that the latter one, supplied with an additional command line, can backtrack generating a graph set $\mathcal{G}$ such that $\mathcal{G}^\diamond \subseteq G_+$, thus outputting minimal hypotheses. With TDAGL one can efficiently compute "weaker" hypotheses, based on a relaxation of the definition of a positive hypothesis that allows for a restricted amount of counterexamples: $|\mathcal{G}^\diamond \cap G_-| \leq k$, where k is a parameter.

Another advantage of TDAGL in comparison with the algorithm described in [20] is that the former can be considered as an algorithm for level-wise construction of projections, starting from smallest ones. To be more precise, TDAGL constructs closures (in the sense of $(\cdot)^{\diamond\diamond}$ operator) of projections, which is more useful. Indeed, consider two graph projections p_1 and p_2 that are subgraphs of same graphs from the graph dataset. In PBRL procedure (see Section 2) these two projections would first correspond to different attributes (they will be clarified only on step 4), thus resulting in the above mentioned drastic growth of the number of attributes. However, in TDAGL the two projections p_1 and p_2 will occur together in the same pattern intent. Actually, the kth level of TDAGL output gives closures of kth projections, the number of which is much smaller than that of kth projections. The graphs comprising these closures may be declared attributes of the representation context, thus making the number of attributes smaller than the number of initial attributes in PBRL. Thus, TDAGL allows one to change the first step of PBRL to

1*. For each example e and for k compute k-projections of $\delta(e)$. The graphs from closures of these projections are declared to be attributes.

5 Conclusions

Lattices arising from descriptions other than sets of attributes are well-known in FCA. Mathematically they are reduced to concept lattices by means of the Basic Theorem of FCA. From the computer science perspective these lattices present problems related to computational complexity.

A method for computing lattices with descriptions given by sets of graphs, starting with rough approximations (low-level projections) was proposed. The algorithm starts from minimal subgraphs and proceeds stepwise using lexicographical order on lists of extents. In contrast to the previous algorithm which constructs the lattice of graph sets bottom-up (starting from object descriptions), this approach allows one to stop at an appropriate level of approximation (projection), thus saving much time and space.

Acknowledgments. This work was supported by the joint COMO project of the Deutsche Forschungsgemeinschaft (DFG) and the Russian Foundation for Basic Research (RFBR).

References

1. Avidon, V.V., Pomerantsev, A.B.: Structure-Activity Relationship Oriented Languages for Chemical Structure Representation. J. Chem. Inf. Comput. Sci. 22(4), 207–214 (1982)
2. Borgelt, C., Berthold, M.R.: Mining Molecular Fragments: Finding Relevant Substructures of Molecules. In: Zhong, N., Yu, P.S. (eds.) Proc. 2nd IEEE International Conference on Data Mining, ICDM 2002, pp. 51–58 (2002)
3. Blinova, V.V., Dobrynin, D.A.: Languages for Representing Chemical Compounds for Intelligent Systems of Chemical Design. Automatic Documentation and Mathematical Linguistics 3 (2000)
4. Blinova, V.G., Dobrynin, D.A., Finn, V.K., Kuznetsov, S.O., Pankratova, E.S.: Toxicology analysis by means of the JSM-method. Bioinformatics 19, 1201–1207 (2003)
5. Cook, D.J., Holder, L.B.: Graph-Based Data Mining. IEEE Intelligent Systems 15(2), 32–41 (2000)
6. Finn, V.K.: Plausible Reasoning in Systems of JSM Type. Itogi Nauki i Tekhniki, Seriya Informatika 15, 54–101 (1991) (in Russian)
7. Ganter, B., Grigoriev, P.A., Kuznetsov, S.O., Samokhin, M.V.: Concept-Based Data Mining with Scaled Labeled Graphs. In: Wolff, K.E., Pfeiffer, H.D., Delugach, H.S. (eds.) ICCS 2004. LNCS (LNAI), vol. 3127, pp. 94–108. Springer, Heidelberg (2004)
8. Gonzalez, J.A., Holder, L.B., Cook, D.J.: Application of Graph-Based Concept Learning to the Predictive Toxicology Domain. In: Helma, C., King, R.D., Kramer, S., Srinivasan, A. (eds.) Proc. Workshop on Predictive Toxicology Challegnge at the 5th Conference on Data Mining and Knowledge Discovery, PKDD 2001, September 6 (2001)
9. Gonzalez, J.A., Holder, L.B., Cook, D.J.: Experimental Comparison of Graph-Based Relational Concept Learning with Inductive Logic Programming Systems. In: Matwin, S., Sammut, C. (eds.) ILP 2002. LNCS (LNAI), vol. 2583, pp. 84–100. Springer, Heidelberg (2003)

10. Ganter, B., Kuznetsov, S.: Formalizing Hypotheses with Concepts. In: Ganter, B., Mineau, G.W. (eds.) ICCS 2000. LNCS (LNAI), vol. 1867, pp. 342–356. Springer, Heidelberg (2000)
11. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: Delugach, H.S., Stumme, G. (eds.) ICCS 2001. LNCS (LNAI), vol. 2120, pp. 129–142. Springer, Heidelberg (2001)
12. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Heidelberg (1999)
13. Grigoriev, P.A., Yevtushenko, S.A.: Elements of an Agile Discovery Environment. In: Grieser, G., Tanaka, Y., Yamamoto, A. (eds.) DS 2003. LNCS (LNAI), vol. 2843, pp. 311–319. Springer, Heidelberg (2003)
14. Inokuchi, A., Washio, T., Motoda, H.: Complete Mining of Frequent Patterns from Graphs: Mining Graph Data. Machine Learning 50(3), 321–354 (2003)
15. King, R.D., Srinivasan, A., Dehaspe, L.: WARMR: A Data Mining tool for chemical data. J. of Computer-Aided Molecular Design 15(2), 173–181 (2001)
16. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. J. Exp. Theor. Artif. Intell. 14(2-3), 189–216 (2002)
17. Kramer, S.: Structural Regression Trees. In: Proc. 13th National Conference on Artificial Intelligence, AAAI 1996, pp. 812–819. AAAI Press/MIT Press (1996)
18. Kuznetsov, S.O.: Similarity operation on hypergraphs as a basis of plausible inference. In: Proc. 1st Soviet Conference on Artificial Intelligence, vol. 1, pp. 442–448 (1988)
19. Kuznetsov, S.O.: JSM-method as a machine learning method. Itogi Nauki i Tekhniki, ser. Informatika 15, 17–50 (1991) (in Russian)
20. Kuznetsov, S.O.: Learning of Simple Conceptual Graphs from Positive and Negative Examples. In: Żytkow, J.M., Rauch, J. (eds.) PKDD 1999. LNCS (LNAI), vol. 1704, pp. 384–391. Springer, Heidelberg (1999)
21. Liquiere, M., Sallantin, J.: Structural Machine Learning with Galois Lattice and Graphs. In: Shavlik, J.W. (ed.) Proc. 15th International Conference on Machine Learning, ICML 1998, pp. 305–313 (1998)
22. Kuznetsov, S.O., Samokhin, M.V.: Learning Closed Sets of Labeled Graphs for Chemical Applications. In: Kramer, S., Pfahringer, B. (eds.) ILP 2005. LNCS (LNAI), vol. 3625, pp. 190–208. Springer, Heidelberg (2005)
23. Nguyen, P.C., Washio, T., Ohara, K., Motoda, H.: Using a Hash-Based Method for Apriori-Based Graph Mining. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., et al. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 349–361. Springer, Heidelberg (2004)
24. Pfahringer, B.: The Futility of Trying to Predict Carcinogenicity of Chemical Compounds. In: Helma, C., King, R.D., Kramer, S., Srinivasan, A. (eds.) Proc. of the Workshop on Predictive Toxicology Challenge at the 5th Conference on Data Mining and Knowledge Discovery, PKDD 2001, September 7 (2001), http://www.predictive-toxicology.org/ptc/
25. Helma, C., King, R.D., Kramer, S., Srinivasan, A. (eds.): Proc. of the Workshop on Predictive Toxicology Challegnge at the 5th Conference on Data Mining and Knowledge Discovery, PKDD 2001, September 7 (2001), http://www.predictive-toxicology.org/ptc/
26. Grigoriev, P.A., Yevtushenko, S.A., Grieser, G.: QuDA, a data miner's discovery environment, FG Intellektik, FB Informatik, Technische Universität Darmstadt, Technical Report AIDA-03-06 (2003), http://www.intellektik.informatik.tu-darmstadt.de/~peter/QuDA.pdf

27. Sebag, M.: Delaying the Choice of Bias: A Disjunctive Version Space Approach. In: Saitta, L. (ed.) Proc. of the 13th International Conference on Machine Learning, pp. 444–452 (1996)
28. Washio, T., Motoda, H.: State of the art of graph-based data mining. SIGKDD Explorations Newsletter 5(1), 59–68 (2003)
29. Yan, X., Han, J.: gSpan: Graph-Based Substructure Pattern Mining. In: Proc. IEEE Int. Conf. on Data Mining, ICDM 2002, pp. 721–724 (2002)
30. Yan, X., Han, J.: CloseGraph: mining closed frequent graph patterns. In: Getoor, L., Senator, T.E., Domingos, P., Faloutsos, C. (eds.) Proc. of the 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, KDD 2003, pp. 286–295. ACM Press, New York (2003)
31. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools with Java Implementations. Morgan Kaufmann, San Francisco (2000)
32. Yan, L.-S.: Study of carcinogenic mechanism of polycyclic aromatic hydrocarbons-extended by region theory and its quantitative model. Carcinogenesis 6(1), 1–6 (1985)
33. Woo, Y.-T., et al.: Use of mechanism-based structure-activity relationships analysis in carcinogenic ranking for drinking water desinfection by-products. Environ. Health Perspect. (1), 75–87 (2002)

Combined Logics of Knowledge, Time, and Actions for Reasoning about Multi-agent Systems

Nikolay V. Shilov and Natalia O. Garanina

Institute of Informatics Systems, Russian Academy of Sciences
6, Lavrentiev ave., 630090, Novosibirsk, Russia
{shilov,garanina}@iis.nsk.su

Abstract. We present a summary of our studies (in period 2002-2007) of the model checking problem for finitely-generated synchronous/asynchronous environments with/without perfect recall for combinations of propositional logics of (common) knowledge, (branching) time, and actions.

1 Introduction

In recent years a surge of interest to applications of modal logics for specification and validation of complex systems became evident. It holds in particular for combined logics of knowledge, time and actions for reasoning about multi-agent systems. In the next paragraphs we explain what are logics of knowledge, time and actions from a viewpoint of mathematicians and philosophers. It provides us a historic perspective and a scientific context for these logics.

For mathematicians and philosophers logics of actions, time, and knowledge can be introduced in few sentences. A logic of actions (ex., EPDL – Elementary Propositional Dynamic Logic [15]) is a polymodal variant of a basic modal logic **K** [3] to be interpreted over arbitrary Kripke models. A logic of time (ex., PLTL – Propositional Linear Temporal Logic [7]) is a modal logic with a number of modalities that correspond to "next time", "always", "sometimes", and "until" to be interpreted in Kripke models over partial orders (discrete linear orders for PLTL in particular). Finally, a logic of knowledge or epistemic logic (ex., PLK – Propositional Logic of Knowledge [8,27]) is a polymodal variant of another basic modal logic **S5** [3] to be interpreted over Kripke models where all binary relations are equivalences. We provide a brief introduction to modal logics in the next section 2 with the aid of Elementary Propositional Dynamic Logic.

Variety of modal logics used for specification of multi-agent systems is due to different properties to be expressed, and (also) due to complexity of *model checking* and *decidability* problems for different logics. *Dynamic logic* and *temporal logics* (and theirs variants with *fixpoints*) are very popular for reasoning about actions and time. Combinations of these logics with logic of knowledge becomes an actual research topic due to naturalness of notions of knowledge and

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 48–58, 2011.

action, and their interactions for reasoning about multi-agent systems. Please refer to section 3 for the definition of background and combined logics.

A number of techniques for (semi)automatic processing of a number of combined logics have been under study. A comprehensive survey of examined logics, techniques, and results is out of scope of the paper. Nevertheless we would like to point out to some relevant research [5,6,8,10,13,14,17,18,23,25,26,33,34].

In contrast, the paper overviews and summarizes our research on model checking problem for pairwise combinations of state-based program and branching-time temporal logics

(1) Elementary Propositional Dynamic Logic (EPDL),
(2) Computation Tree Logic extended by actions (*Act*-CTL),
(3) the propositional μ-Calculus (μC)

with the following epistemic logics

(a) Propositional Logic of Knowledge for n agents (PLK$_n$),
(b) Propositional Logic of Common Knowledge for n agents (PLC$_n$).

Results of these studies are presented in the section 4. Previously they (altogether with formal proofs) have been published by parts in a technical report [29], in proceedings [30,11,32,12], and in a journal [31].

2 Elements of Modal Logic

All modal logics are languages that are characterized by syntax and semantics. Let us define below a very simple modal logic in this way. This logic is called Elementary Propositional Dynamic Logic (EPDL).

Let *true*, $false$ be Boolean constants, *Prp* and *Rel* be disjoint sets of propositional and relational variables respectively. The syntax of the classical propositional logic consists of formulas which are constructed from propositional variables and Boolean connectives '$\neg$' (negation), '$\wedge$' (conjunction), '$\vee$' (disjunction), '$\rightarrow$' (implication), and '$\leftrightarrow$' (equivalence) in accordance with the standard rules. EPDL has additional formula constructors, modalities, which are associated with relational variables: if r is a relational variable and ϕ is a formula of EPDL then

- $([r]\phi)$ is a formula which is read as "box r-ϕ" or "after r always ϕ";
- $(\langle r\rangle\phi)$ is a formula which is read as "diamond r-ϕ" or "after r sometimes ϕ".

The semantics of EPDL is defined in models, which are called Labeled Transition Systems (LTS) by Computer Scientists and Kripke[1] models by mathematicians and philosophers. A model M is a pair (D, I) where the domain (or the universe) D is a set, while the interpretation I is a pair of mappings (P, R). Elements of

[1] Due to pioneering papers of Saul Aaron Kripke (born in 1940) on models for modal logics.

the domain D are called states by Computer Scientists and worlds by mathematicians and philosophers. The interpretation maps propositional variables to sets of states $P : Prp \to 2^D$ and relational variables to binary relations on states $R : Rel \to 2^{D \times D}$. We write $I(p)$ and $I(r)$ instead of $P(p)$ and $R(r)$ whenever it is implicit that p and r are propositional and relational variables respectively.

Every model $M = (D, I)$ can be viewed as a directed graph (infinite maybe) with nodes and edges labeled by propositional and action variables respectively. Its nodes are states of D. A node $s \in D$ is marked by a propositional variable $p \in Prp$ iff $s \in I(p)$. A pair of nodes $(s1, s2) \in D \times D$ is an edge of the graph iff $(s1, s2) \in I(r)$ for some relational variable $r \in Rel$; in this case the edge $(s1, s2)$ is marked by this relational variable r. Conversely, a graph with nodes and edges labeled by propositional and relational variables respectively can be considered as a model.

For every model $M = (D, I)$ the entailment (validity, satisfiability) relation '$\models_M$' between states and formulas can be defined by induction on formula structure: for every state $s \in D$

- $s \models_M true$ and not $s \models_M false$;
- for any propositional variable p, $s \models_M p$ iff $s \in I(p)$;
- for any formula ϕ, $s \models_M (\neg\phi)$ iff it is not the case $s \models_M \phi$;
- for any formulas ϕ and ψ,
  - $s \models_M (\phi \wedge \psi)$ iff $s \models_M \phi$ and $s \models_M \psi$;
  - $s \models_M (\phi \vee \psi)$ iff $s \models_M \phi$ or $s \models_M \psi$;
- for any relational variable r, and formula ϕ,
  - $s \models_M ([r]\phi)$ iff $(s, t) \in I(r)$ implies $t \models_M \phi$ for every state t;
  - $s \models_M (\langle r\rangle\phi)$ iff $(s, t) \in I(r)$ and $t \models_M \phi$ for some state t.

Semantics of the above kind is called possible worlds semantics.

Let us explain EPDL pragmatics by the following puzzle example. Alice and Bob play the Number Game. Positions in the game are integers in [1..109]. An initial position is a random number. Alice and Bob make alternating moves: Alice, Bob, Alice, Bob, etc. Available moves are the same for both: if a current position is $n \in$[1..99] then $(n + 1)$ and $(n + 10)$ are possible next positions. A player wins the game iff the opponent is the first to enter [100..109]. The problem: to find all initial positions where Alice has a winning strategy.

Kripke model for the game is quite obvious:

- States correspond to game positions, i.e. integers in [1..109].
- Propositional variable $fail$ is interpreted by [100..109].
- Relational variable $move$ is interpreted by possible moves.

Formula $\neg fail \wedge \langle move\rangle(\neg fail \wedge [move]fail)$ is valid in those states where the game is not lost, and there exists a move after which the game is not lost, and then all possible moves always lead to a loss in the game. Hence this EPDL formula is valid in those states where Alice has a 1-round winning strategy against Bob.

3 Combining Knowledge, Actions and Time

3.1 Propositional Logic for Epistemic Agents

Logics for reasoning about knowledge are also known as epistemic logics. One of the simplest epistemic logics is Propositional Logic of Knowledge for $n > 0$ agents (PLK_n) is defined below. Please refer to [8] for the definition of the more complicated Propositional Logic of Common knowledge for $n > 0$ agents (PLC_n).

A special terminology, notation and Kripke models are used in this framework. A set of relational symbols *Rel* in PLK_n consists of natural numbers [1..n] representing names of agents. Notation for modalities is: if $i \in [1..n]$ and ϕ is a formula, then $(K_i\phi)$ and $(S_i\phi)$ are used instead of $([i]\phi)$ and $(\langle i\rangle\phi)$. These formulas are read as "(an agent) i knows ϕ" and "(an agent) i can suppose ϕ". For every agent $i \in$[1..n] in every model $M = (D, I)$, interpretation $I(i)$ is an "indistinguishability relation", i.e. an equivalence relation between states that the agent i can not distinguish. Every model M, where all agents are interpreted in this way, is denoted as $(D, \sim_1, ... \sim_n, I)$ with explicit $I(1) = \sim_1$, ... $I(n) = \sim_n$ instead of the brief standard notation (D, I). An agent knows some "fact" ϕ in a state s of the model M, if the fact is valid in every state t of this model that the agent can not distinguish from s: $s \models_M (K_i\phi)$ iff $t \models_M \phi$ for every state t such that $s \sim_i t$. Similarly, an agent can suppose a "fact" ϕ in a state s of the model M, if the fact is valid in some state t of this model that the agent can not distinguish from s: $s \models_M (S_i\phi)$ iff $t \models_M \phi$ for some state t such that $s \sim_i t$.

The above possible worlds semantics of knowledge has been introduced in [16]. A philosophical "motivation" behind it goes back to Plato who defined "knowledge as true belief", i.e. a personal (by an "agent") perception ("belief") of the unverse that holds in the current state of the world (i.e. "is true"). Please refer to [16,8,27] for more information.

3.2 Branching Temporal Logic with Actions

Another propositional polymodal logic is Computational Tree Logic with Actions (*Act*-CTL). *Act*-CTL is a variant of a basic propositional branching time temporal logic, namely Computational Tree Logic (CTL) [7,4]. In *Act*-CTL the set of relational symbols consists of action symbols *Act*. Each action symbol can be interpreted by an "instant action" that is executable in one "undividable" moment of time.

Act-CTL notation for basic modalities is: if $b \in Act$ and ϕ is a formula, then $(\mathbf{A}_b\mathbf{X}\phi)$ and $(\mathbf{E}_b\mathbf{X}\phi)$ are used instead of $([b]\phi)$ and $(\langle b\rangle\phi)$. But syntax of *Act*-CTL has also some other special constructs associated with action symbols: if $b \in Act$ and ϕ and ψ are formulas, then $(\mathbf{A}_b\mathbf{G}\phi)$, $(\mathbf{A}_b\mathbf{F}\phi)$, $(\mathbf{E}_b\mathbf{G}\phi)$, $(\mathbf{E}_b\mathbf{F}\phi)$, $\mathbf{A}_b(\phi\mathbf{U}\psi)$ and $\mathbf{E}_b(\phi\mathbf{U}\psi)$ are also formulas of *Act*-CTL. In formulas of *Act*-CTL prefix '**A**' is read as "for every run", '**E**' – "for some run", a sub-index '*b*' is read as "in *b*-run(s)", and stem '**X**' is read as "next state", '**G**' – "always" or "globally", '**F**' – "sometimes" or "future", '**U**' – "until".

We have already explained semantics of $(\mathbf{A}_b\mathbf{X}\phi)$ and $(\mathbf{E}_b\mathbf{X}\phi)$ by referencing to $([b]\phi)$ and $(\langle b\rangle\phi)$. Constructs '$\mathbf{A}_b\mathbf{G}$', '$\mathbf{A}_b\mathbf{F}$', '$\mathbf{E}_b\mathbf{G}$', and '$\mathbf{E}_b\mathbf{F}$' can be expressed in terms of '$\mathbf{A}_b(\dots\mathbf{U}\dots)$' and '$\mathbf{E}_b(\dots\mathbf{U}\dots)$', for example: $(\mathbf{E}_b\mathbf{G}\phi) \equiv (\mathbf{E}_b(true\mathbf{U}\phi)$. (Please, refer to [31] for details.) Thus let us define below semantics of '$\mathbf{A}_b(\dots\mathbf{U}\dots)$' and '$\mathbf{E}_b(\dots\mathbf{U}\dots)$' only.

Let $M = (D, I)$ be a model. If $b \in Act$ is an action symbol, then a partial b-run is a sequence of states $s_0, \dots s_k, s_{(k+1)}, \dots \in D$ (maybe infinite) such that $(s_k, , s_{(k+1)}) \in I(b)$ for every consecutive pair of states within this sequence; a b-run is an infinite partial b-run or a finite b-run that can not be continued. Then semantics of constructs '$\mathbf{A}_b(\dots\mathbf{U}\dots)$' and '$\mathbf{E}_b(\dots\mathbf{U}\dots)$' can be defined as follows:

- $s \models_M (\mathbf{A}_b(\phi\mathbf{U}\psi)$ iff for every b-run $s_0, \dots s_k, \dots$ that starts in s (i.e. $s_0 = s$) there exists some $n \geq 0$ for which $s_n \models_M \psi$ and $s_k \models_M \phi$ for every $k \in [0..(n-1)]$;
- $s \models_M (\mathbf{E}_b(\phi\mathbf{U}\psi)$ iff for some b-run $s_0, \dots s_k, \dots$ that starts in s (i.e. $s_0 = s$) there exists some $n \geq 0$ for which $s_n \models_M \psi$ and $s_k \models_M \phi$ for every $k \in [0..(n-1)]$.

The standard branching-time temporal logic CTL can be treated as *Act*-CTL with a single implicit action symbol for a tick of clocks. As follows from the above, CTL and EPDL can be expressed in *Act*-CTL. In the next section 4 we use a so-called propositional μ-Calculus (μC) of D. Kozen [20,22] that is more expressive than *Act*-CTL. Please refer [28] for a gentle introduction of μC.

3.3 Combined Logics of Knowledge, Actions, and Time

There are many combined polymodal logics for reasoning about multi-agent systems. Maybe the most advanced is Belief-Desire-Intention (BDI) logic [33,34]. An agent's beliefs correspond to information the agent has about the world. (This information may be incomplete or incorrect; in BDI agent's knowledge is just a true belief.) An agent's intensions correspond to the allocated tasks. An agent's desires represent what it has committed to achieve. Admissible actions are actions of individual agents; they may be constructed from primitive actions by means of composition, non-deterministic choice, iteration, and parallel execution. But semantics of BDI and reasoning in BDI are quite complicated.

In contrast, let us discuss below a simple example of a combined logic of knowledge, actions and time – namely Propositional Logic of Knowledge and Branching Time for $n > 0$ agents *Act*-CTL-K$_n$. For a formal definition of other combined logics that are in use in this paper, please, refer to [11,31,32].

First we provide a formal definition of *Act*-CTL-K$_n$, then discuss some pragmatics, and then – in the next section 4 – introduce and study model checking as a reasoning mechanism.

Let [1..n] be a set of agents ($n > 0$), and *Act* be a finite alphabet of action symbols. Syntax of *Act*-CTL-K$_n$ admits epistemic modalities K_i and S_i for every $i \in [1..n]$, and branching-time constructs $\mathbf{A}_b\mathbf{X}$, $\mathbf{E}_b\mathbf{X}$, $\mathbf{A}_b\mathbf{G}$, $\mathbf{E}_b\mathbf{G}$, $\mathbf{A}_b\mathbf{F}$, $\mathbf{E}_b\mathbf{F}$, $\mathbf{A}_b(\dots\mathbf{U}\dots)$, and $\mathbf{E}_b(\dots\mathbf{U}\dots)$ for every $b \in Act$. Semantics is defined

in terms of entailment in environments. An (epistemic) environment is a tuple $E = (D, \sim_1, \ldots \sim_n, I)$ that is a model for PLK$_n$ and for *Act*-CTL simultaneously. Entailment relation $\models_E$ is defined by induction according to the standard definition for propositional connectives (see semantics of EPDL), and the above definitions of epistemic modalities and branching time constructs.

We are mostly interested in trace-based perfect recall synchronous environments generated from background finite environments. "Finitely generated trace-based" means that possible "worlds" are runs of finite-state machine(s). There are several opportunities how to define semantics of combined logics on runs. In particular, there are two extreme cases: Forgetful Asynchronous Systems (FAS) and Synchronous systems with Perfect Recall (PRS). "Perfect recall" means that every agent has a log-file with all his/her observations along a run, while "forgetful" means that information of this kind is not available. "Synchronous" means that every agent can distinguish runs of different lengths, while "asynchronous" means that some runs of different lengths may be indistinguishable.

4 Model Checking Problem for Combined Logics

It is quite natural that in the FAS case combined logics *Act*-CTL-K$_n$ can express as much as it can express in the background finite system. In contrast, in the PRS case *Act*-CTL-K$_n$ becomes much more expressive than in the background finite environment. Importance of combined logics in the framework of trace-based semantics with synchronous perfect recall rely upon their characteristic as logics of agent's learning or knowledge acquisition. We would like to argue this characteristic by the following single-agent Fake Coin Puzzle $FCP(N, M)$.

- A set consists of $(N + 1)$ enumerated coins. The last coin is a valid one. A single coin with a number in $[1..N]$ is fake, but all other coins with numbers in $[1..N]$ are valid. All valid coins have the same weight that differs from the weight of the fake. Is it possible to identify the fake by balancing coins M times at most?

In $FCP(N, M)$ the agent (i.e. a person who has to solve the puzzle) does not know neither a number of the fake, nor whether it is lighter or heavier than the valid coins. Nevertheless, this number is in $[1..N]$, and the fake coin is either lighter (l) or heavier (h). The agent can make balancing queries and read balancing results after each query. Every balancing query is an action $b(L, R)$ which consists in balancing of two disjoint sets of coins: with numbers $L \subseteq [1..N + 1]$ on the left pan, and with numbers $R \subseteq [1..N + 1]$ on the right pan, $|L| = |R|$. There are three possible balancing results: '$<$', '$>$', and '$=$', which means that the left pan is lighter, heavier than or equal to the right pan, respectively. Of course, there are initial states (marked by ini) which represent a situation when no query has been made.

Let us summarize. The agent acts in the environment generated from a finite space $[1..N] \times \{l, h\} \times \{<, >, =, ini\}$. His/her admissible actions are balancing queries $b(L, R)$ for disjoint $L, R \subseteq [1..N+1]$ with $|L| = |R|$. The only information

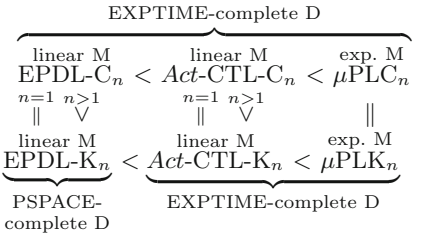


Fig. 1. Logic fusion summary

available for the agent (i.e., which gives him/her an opportunity to distinguish states) is a balancing result. The agent should learn $fake\_coin\_number$ from a sequence which may start from any initial state and then consists of M queries and corresponding results. Hence a single agent logic *Act*-CTL-K$_1$ seems to be a very natural framework for expressing $FCP(N, M)$ as follows: to validate or refute whether

$$s \models_E (\underbrace{\mathbf{E}_B\mathbf{X}\ldots\mathbf{E}_B\mathbf{X}}_{M\ times} \bigvee_{f\in[1..N]} K_1(fake\_coin\_number = f))$$

for every initial state s, where E is the PRS environment generated from a $[1..N]\times\{l,h\}\times\{<,>,=,ini\}$, and B is a balancing query $\bigcup_{L,R\subseteq[1..N+1]} b(L,R)$.

The model checking problem for a combined logic (*Act*-CTL-K$_n$ in particular) and a class of epistemic environments (ex., PRS or FAS environments) is to validate or refute $s \models_E \phi$, where E is a finitely-generated environment in the class, s is an 'initial state' of the environment E, and ϕ is a formula of the logic. The above re-formulation of $FCP(N, M)$ is a particular example of a model checking problem for a formula of *Act*-CTL-K$_n$ and some finitely-generated perfect recall environment.

In particular, papers [29,30,11] have examined model checking as well as the expressive power and the decidability for combinations of listed logics in Kripke semantics. These results are summarized in Fig. 1. Notation adopted in this figure can be explained as follows. "PSPACE/EXTIME-complete D" means that the decidability problem is PSPACE/EXTIME-complete respectively. Notation "linear/exp. M" means that there exists a model checker with linear or exponential time complexity. (In)equalities stay for expressive power. Parameter n stays for the number of agents in the multi-agent system.

It is quite natural that in the FAS case neither of the discussed combined logics can express more than it can express in a background finite system. In other words, for formulae of these logics every system is an abstraction of forgetful asynchronous traces generated by this system. It implies that all results represented in Fig. 1 remain valid for the FAS case.

In contrast, in the PRS case the model checking problem becomes much more complicated than in background finite systems. We have demonstrated that if

the number of agents n in a multi-agent system is $n > 1$ then the model checking problem in perfect recall synchronous semantics

- is $PSPACE$-complete for EPDL-C$_n$,
- has non-elementary upper and lower bounds for Act-CTL-K$_n$,
- is undecidable for Act-CTL-C$_n$, μPLK$_n$ and μPLC$_n$.

These results correlate with [25] where model checking problem for synchronous systems with perfect recall and fusion of PLK$_n$ and PLC$_n$ with Propositional Logic of Linear Time (PLLT) has been examined. The cited paper has demonstrated that model checking for synchronous systems with perfect recall in the two agents case ($n = 2$)

- is $PSPACE$-complete for UNTIL-free PLLT-C$_n$,
- has non-elementary upper and lower bounds for PLLT-K$_n$,
- is undecidable for PLLT-C$_n$.

We should remark that our results and the results presented in [25] are closely related to [24], where $PSPACE$-completeness and undecidability have been proved for model checking formulae of PLC$_n$ in synchronous and asynchronous systems with perfect recall. The paper [25] and the present paper extend time-free results from [24] for linear/branching time respectively.

Paper [25] also has suggested a tree-like data structures for model checking of linear time and knowledge with bounded nesting. This abstraction is very convenient for representation of knowledge evolution and update. It comprises the trees whose depth is equal to knowledge nesting [24]. Paper [25] has demonstrated that the model checking problem for PLLT-K$_n$ in the synchronous perfect recall semantics can be reduced to the problem of emptiness of Büchi automaton whose inputs are infinite sequences of these trees.

We have adopted similar trees for model checking Act-CTL-K$_n$ in PRS semantics in [31]. The cited paper has presented a "direct" (update+abstraction)-algorithm for model checking Act-CTL-K$_n$ in perfect recall synchronous environments. The algorithm is based on a transformation of Act-CTL-K$_n$ formulas into formulas of Act-CTL, and on a reduction of an infinite synchronous perfect recall system to a finite model TR_k which consists of k-trees (special finite trees of height k). The complexity of the algorithm is given by the following proposition.

Proposition 1
For every integer $k \geq 1$ and $n \geq 1$, synchronous environment with perfect recall $PRS(E)$, every formula ϕ of Act-CTL-K$_n$ with the knowledge depth k at most, the model checking problem is decidable with the upper bound

$$O\Big(f \times \frac{exp(n \times d, k) \times (exp(n \times d, k-1))^2}{n^3}\Big),$$

where f is the size of the formula, d is the number of states in D_E, and the function $exp(a, b)$ is

$$exp(a, b) = \begin{cases} a, & \text{if } b = 0, \\ a \times 2^{exp(a,b-1)}, & \text{otherwise.} \end{cases}$$

Unfortunately, the upper bound for the size of this finite model is a non-elementary function of the number of states [31]. Hence a straightforward use of a model checker for CTL for model checking *Act*-CTL on k-trees is likely to be a non-feasible task. Roughly speaking, this space is too big to be treated as finite. It implies that it makes sense to try to apply techniques which have been developed for infinite-state model checking, for model checking Act^{+n}-CTL on k-trees.

A very popular approach to infinite-state model checking is a formalism of well-structured labeled transition systems [1,9]. Foundational papers [1,9] have proved the decidability of liveness (reachability) and progress (eventuality) properties in well-structured single action labeled transition systems. Roughly speaking, a well-structured single action labeled transition system is provided with a (pre-)order, and its transitions 'preserve' this (pre-)order, and its labeling forms cones with respect to this (pre-)order. Paper [21] has generalized cited decidability results for disjunctive formulae of the propositional μ-Calculus [20,2] in well-structured labeled multi-action transition systems. Model checking of disjunctive properties in well-structured labeled transition systems computes finite bases of cones of states that enjoy the property.

In the paper [32] we have demonstrated that the model TR_k provided with a sub-tree partial order forms a well-structured labeled transition system where every property expressible in the μ-Calculus, can be characterized by a finite computable set of maximal trees that enjoy the property. We tried feasibility of this approach to model checking of *Act*-CTL-K$_n$ in trace-based perfect recall synchronous environment by automatic model checking a simple, but "huge" example[2].

Acknowledgements. Research has been supported by joint grant RFBR 05-01-04003-a-DFG project COMO, GZ: 436 RUS 113/829/0-1, and currently is supported in parts by Integration Program 2/12 of Russian Academy of Science and by grant RFBR 10-01-00532-a.

References

1. Abdulla, P.A., Ĉerâns, K., Jonsson, B., Tsay, Y.-K.: Algorithmic analysis of programs with well quasi-ordered domains. Information and Computation 160(1-2), 109–127 (2000)
2. Arnold, A., Niwinski, D.: Rudiments of μ-calculus. North Holland, Amsterdam (2001)
3. Bull, R., Segerberg, K.: Basic Modal Logic. In: Gabbay, D., Cuenthner, F. (eds.) Handbook of Philosophical Logic, vol. 3. Kluwer Academic Publishers, Dordrecht (2001)
4. Clarke, E., Grumberg, O., Peled, D.: Model Checking. MIT Press, Cambridge (1999)
5. Dixon, C., Fernandez Gago, M.-C., Fisher, M., van der Hoek, W.: Using Temporal Logics of Knowledge in the Formal Verification of Security Protocols. In: Proceedings of TIME 2004, Tatihou, Normandie, France, July 1-3. IEEE, Los Alamitos (2004)

[2] The size of the initial environment is 120000 and the size of the generated finite model is about 10^{36000}.

6. Dixon, C., Nalon, C., Fisher, M.: Tableau for Logics of Time and Knowledge with Interactions Relating to Synchrony. Journal of Applied Non-Classical Logics 14(4), 397–445 (2004)
7. Emerson, E.A.: Temporal and Modal Logic. In: van Leeuwen, J., Meyer, A.R., Nivat, M., Paterson, M., Perrin, D. (eds.) Handbook of Theoretical Computer Science (B). Elsevier, The MIT Press (1990)
8. Fagin, R., Halpern, J.Y., Moses, Y., Vardi, M.Y.: Reasoning about Knowledge. MIT Press, Cambridge (1995)
9. Finkel, A., Schnoebelen, P.: Well-structured transition systems everywhere! Theor. Comp. Sci. 256(1-2), 63–92 (2001)
10. Gammie, P., van der Meyden, R.: MCK: Model checking the logic of knowledge. In: Alur, R., Peled, D.A. (eds.) CAV 2004. LNCS, vol. 3114, pp. 479–483. Springer, Heidelberg (2004)
11. Garanina, N.O., Kalinina, N.A., Shilov, N.V.: Model checking knowledge, actions and fixpoints. In: Proc. of Concurrency, Specification and Programming Workshop CS&P 2004, Germany, Humboldt Universitat, Berlin, Informatik-Bericht, vol. 2(170), pp. 351–357 (2004)
12. Garanina, N.O., Shilov, N.V.: Model Checking Knowledge of acting Agents with log-files. In: Meetings of Minds: Proc. of Int. Workshop Logic, Rationality and Interaction, China (2007)
13. Halpern, J.Y., Vardi, M.Y.: The complexity of Reasoning About Knowledge and Time. In: Proc. of Symp. Theor. of Computing (STOC), pp. 304–315 (1986)
14. Halpern, J.Y., van der Meyden, R., Vardi, M.Y.: Complete Axiomatizations for Reasoning about Knowledge and Time. SIAM Journal on Computing 33(3), 674–703 (2004)
15. Harel, D., Kozen, D., Tiuryn, J.: Dynamic Logic. MIT Press, Cambridge (2000)
16. Hintikka, J.: Knowledge and Belief. Cornell University Press, Ithica (1962)
17. van der Hoek, W., Wooldridge, M.J.: Model Checking Knowledge and Time. In: Bošnački, D., Leue, S. (eds.) SPIN 2002. LNCS, vol. 2318, pp. 95–111. Springer, Heidelberg (2002)
18. Kacprzak, M., Lomuscio, A., Penczek, W.: Unbounded Model Checking for Knowledge and Time. In: Proceedings of the CS&P 2003 Workshop, Warsaw University, vol. 1, pp. 251–264 (2003)
19. Kacprzak, M., Penczek, W.: Model Checking for Alternating-Time mu-Calculus via Translation to SAT. In: Proc. of Concurrency, Specification and Programming Workshop CS&P 2004, Germany, Humboldt Universitat, Berlin, Informatik-Bericht, vol. 2(170) (2004)
20. Kozen, D.: Results on the Propositional Mu-Calculus. Theoretical Computer Science 27(3), 333–354 (1983)
21. Kouzmin, E.V., Shilov, N.V., Sokolov, V.A.: Model Checking μ-Calculau in Well-Structured Transition Systems. In: Proceedings of 11th International Symposium on Temporal Representation and Reasoning (TIME 2004), France, pp. 152–155. IEEE Press, Los Alamitos (2004)
22. Kozen, D., Tiuryn, J.: Logics of Programs. In: Handbook of Theoretical Computer Science, vol. B, pp. 789–840. Elsevier, MIT Press (1990)
23. Lomuscio, A., Penczek, W.: Verifying Epistemic Properties of Multi-agent Systems via Bounded Model Checking. Fundamenta Informaticae 55(2), 167–185 (2003)
24. van der Meyden, R.: Common Knowledge and Update in Finite Environments. Information and Computation 140(2), 115–157 (1998)

25. van der Meyden, R., Shilov, N.V.: Model checking knowledge and time in systems with perfect recall. In: Pandu Rangan, C., Raman, V., Sarukkai, S. (eds.) FST TCS 1999. LNCS, vol. 1738, pp. 432–445. Springer, Heidelberg (1999)
26. van der Meyden, R., Wong, K.: Complete Axiomatizations for Reasoning about Knowledge and Branching Time. Studia Logica 75(1), 93–123 (2003)
27. Rescher, N.: Epistemic Logic. Survey of the Logic of Knowledge. University of Pittsburgh Press, Pittsburgh (2005)
28. Shilov, N.V., Yi, K.: How to find a coin: propositional program logics made easy. In: Current Trends in Theoretical Computer Science, vol. 2, pp. 181–213. World Scientific, Singapore (2004)
29. Shilov, N.V., Garanina, N.O.: Combining Knowledge and Fixpoints. Preprint n. 98 of A.P. Ershov Institute of Informatics Systems, Novosibirsk (2002)
30. Shilov, N.V., Garanina, N.O.: Model Checking Knowledge And Fixpoints. In: Proc. 4th Int. Workshop on Fixed Points on Computer Science, Copenhagen, Denmark, pp. 25–39 (2002)
31. Shilov, N.V., Garanina, N.O., Choe, K.-M.: 2.7. Update and Abstraction in Model Checking of Knowledge and Branching Time. Fundameta Informaticae 72(1-3), 347–361 (2006)
32. Shilov, N.V., Garanina, N.O.: Well-Structured Model Checking of Multiagent Systems. In: Virbitskaite, I., Voronkov, A. (eds.) PSI 2006. LNCS, vol. 4378, pp. 363–376. Springer, Heidelberg (2007)
33. Wooldridge, M.: Practical reasoning with procedural knowledge: A logic of BDI agents with know-how. In: Gabbay, D.M., Ohlbach, H.J. (eds.) FAPR 1996. LNCS, vol. 1085, pp. 663–678. Springer, Heidelberg (1996)
34. Wooldridge, M.: An Introduction to Multiagent Systems. John Wiley & Sons Ltd., Chichester (2002)

Applications of Temporal Conceptual Semantic Systems*

Karl Erich Wolff

Mathematics and Science Faculty
Darmstadt University of Applied Sciences
Holzhofallee 38, D-64295 Darmstadt, Germany
karl.erich.wolff@t-online.de

Abstract. Based on Formal Concept Analysis the notion of a Temporal Conceptual Semantic System is introduced as a formal conceptual representation for temporal systems with arbitrary discrete or continuous semantic scales. In this paper, we start with an example of a weather map with a moving high pressure zone to explain the basic notions for Temporal Conceptual Semantic Systems. The central philosophical notion of an object is represented as a formal concept or, more flexible, as a tuple of concepts. Generalizing the idea of a volume of an object in physics we introduce the notion of a *trace* of an object in some space. This space is described as a continuous or discrete concept lattice. Combining the notion of a trace of an object with the notion of a time granule yields the notion of a state of an object at some time granule. This general notion of a state allows for a clear conceptual understanding of particles, waves and Heisenberg's Uncertainty Relation. Besides these theoretical aspects, Temporal Conceptual Semantic Systems can be used very effectively in practice. That is shown for data of a distillation column using a nested transition diagram.

1 Introduction

In this paper we introduce a simple and powerful conceptual framework which is closely connected to many quite different fields of classical and modern knowledge representation. This framework has been developed in Conceptual Knowledge Processing which is based on Formal Concept Analysis [18, 10]. First ideas for this framework have been developed in applications of conceptual scaling in Fuzzy Theory [37, 38, 25], Rough Set Theory [14, 21], and General Systems Theory [5, 11, 12, 13, 20]. One of the basic intuitive notions in General Systems Theory is the notion of a state, but the formalization of this notion leads to many difficulties. We cite L. Zadeh [36]:

> *To define the notion of a state in a way which would make it applicable to all systems is a difficult, perhaps impossible, task.*

* Supported by DFG project COMO, GZ: 436 RUS 113/829/0-1.

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 59–78, 2011.

The notion of a state was defined in a first conceptual version by the author in [20] and generalized in several steps to the actual notion of a state in a Temporal Conceptual Semantic System in this paper. During this development a common generalization of the notion of states in physics [8, 6, 7, 27] and in automata theory [1, 9, 24] had been established. For a long time it was not clear how to define a conceptual notion of a *state* such that the quantum theoretical states which possess a probability distribution are also describable. The main step into that direction was the introduction of Conceptual Semantic Systems [26] which reached the aim to find a common conceptual description of particles and waves in physics.

Based on the notion of *aspects* a first conceptual representation of Heisenberg's Uncertainty Relation has been presented in [29]. This has been developed further in [30] where a more general *object* notion, the *instance selection* and an explicit representation of *measurements* has been introduced. In the paper at hand the basic notions in Temporal Concept Analysis as developed by the author in previous articles [22, 23, 26, 27, 28, 29, 30] are generalized and clarified by the notion of a *Temporal Conceptual Semantic System* (TCSS). That is based on the notion of a *Conceptual Semantic System* (CSS) where two new tools are introduced, namely the formal representation of objects by *tuples of concepts* of the semantic scales and the *instance selection* which assigns to each tuple of concepts a "relevant" subset of the set of instances. In the following section we use a moving high pressure zone as a typical example of a *distributed* temporal object in a TCSS. We explain this special TCSS starting with a data table and interpreting the values of the data table as formal concepts of suitable formal contexts. For readers who are not familiar with the notions in Formal Concept Analysis [10] we introduce some formal contexts and their concept lattices in this example of a moving high pressure zone. Later on we will introduce the mathematical definitions around TCSSs.

2 Example: A Moving High Pressure Zone

As a small example we construct a TCSS which yields a weather map with a moving high pressure zone over Germany. To keep the data table small we construct a map of Germany using a coarse grid of longitude and latitude coordinates into which we embed 15 towns. To represent a moving high pressure zone we assume that the pressure has been measured in some weather stations (WS) at two consecutive days, say Monday and Tuesday. The data are shown in Tab.1 where the rows 1,...,15 show the latitude and longitude values of 15 German towns, the rows 16,...,25 show the pressure values (in hectopascal (hPa)) measured at certain days at certain weather stations located in some of the previously mentioned towns.

In Table 1 the row labeled by *instance* 1 tells that the *place Berlin* has *latitude* 52.5 and longitude 13.4 and no time and no pressure is recorded in this line, shown by the sign "/" in the column for *time* and *pressure*. *Instance* 16 tells that *Weather Station Dortmund* has *latitude* 51.5 and *longitude* 7.5 and has reported at *Monday* a *pressure* of 1020 hectopascal.

Table 1. Data table of a Moving High Pressure Zone

instance	place	latitude	longitude	time	pressure
1	Berlin	52.5	13.4	/	/
2	Dortmund	51.5	7.5	/	/
3	Dresden	51.0	13.7	/	/
4	Frankfurt (Main)	50.1	8.7	/	/
5	Frankfurt (Oder)	52.3	14.5	/	/
6	Freiburg	48.0	7.9	/	/
7	Hamburg	53.6	10.0	/	/
8	Kassel	51.3	9.5	/	/
9	Köln	51.0	7.0	/	/
10	Magdeburg	52.1	11.6	/	/
11	München	48.1	11.6	/	/
12	Nürnberg	49.5	11.1	/	/
13	Passau	48.6	13.5	/	/
14	Saarbrücken	49.2	7.0	/	/
15	Wilhelmshaven	53.5	8.1	/	/
16	WS Dortmund	51.5	7.5	Monday	1020
17	WS Frankfurt (Main)	50.1	8.7	Monday	1030
18	WS Hamburg	53.6	10.0	Monday	970
19	WS Kassel	51.3	9.5	Monday	1010
20	WS München	48.1	11.6	Monday	980
21	WS Berlin	52.5	13.4	Tuesday	1020
22	WS Dresden	51.1	13.7	Tuesday	1010
23	WS Frankfurt (Oder)	52.3	14.5	Tuesday	1030
24	WS Hamburg	53.6	10.0	Tuesday	970
25	WS München	48.1	11.6	Tuesday	980

2.1 Basic Conceptual Notions

For the conceptual representation of knowledge as it is contained in Table 1 we interpret the values in each column as elements of a conceptual hierarchy which is formally represented as the concept lattice of a formal context. In Formal Concept Analysis [10] a formal context $\mathbb{K}$ is defined as a triple (G, M, I) of sets such that $I \subseteq G \times M$. The set G is called the set of formal objects ("Gegenstände"), M is called the set of formal attributes ("Merkmale"), and the binary relation I is called the incidence relation of the formal context (G, M, I). For $g \in G$ and $m \in M$ we write "gIm" instead of "$(g, m) \in I$". A formal concept of a formal context (G, M, I) is a pair (A, B) where $A \subseteq G$, $B \subseteq M$, and $A = \{g \in G \mid \forall_{m \in B} gIm\}$ and $B = \{m \in M \mid \forall_{g \in A} gIm\}$. The set A is called the *extent*, and B the *intent* of the formal concept (A, B). The set $\mathfrak{B}(\mathbb{K})$ of all formal concepts of a formal context $\mathbb{K}$ is ordered by: $(A, B) \leq (C, D) :\Leftrightarrow A \subseteq C (\Leftrightarrow B \supseteq D)$. The ordered set $(\mathfrak{B}(\mathbb{K}), \leq)$ is a complete lattice, called the *concept lattice* of $\mathbb{K}$. For each formal object $g \in G$ the smallest concept containing g in its extent is called the object concept $\gamma(g)$ of g. Dually

the attribute concept $\mu(m)$ of a formal attribute m is defined. It can be proved easily that $\forall_{g\in G, m\in M} gIm \Leftrightarrow \gamma(g) \leq \mu(m)$.

2.2 The Scales of Our Example

In our example of a moving high pressure zone we represent the values in the column for time as formal concepts of the formal context $\mathbb{S}_{time}$, called the semantic scale for time, shown in Table 2.

Table 2. The time scale $\mathbb{S}_{time}$

$\mathbb{S}_{time}$	Monday	Tuesday	days
Monday	×		×
Tuesday		×	×
/			

In the formal context $\mathbb{S}_{time}$ the "missing value" "/" is treated as a formal object without any attribute. The concept lattice $\mathfrak{B}(\mathbb{S}_{time})$ is shown in Fig. 1. The arrow in Fig. 1 represents the time relation which is used to define transitions; that will be discussed later.

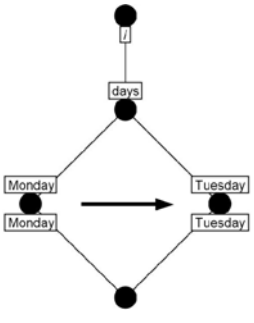


Fig. 1. The semantic scale for *time* with time relation

Similarly to the column for time we choose semantic scales for the other four columns of Table 1 which are different from the instance column; the instance column shows just a labeling of the rows.

The semantic scale $\mathbb{S}_p$ for *pressure* is defined as a modified interordinal scale on the multiples of 10 in the interval $[970, 1030]$. Its concept lattice is shown in Fig. 2.

In this interordinal scale for pressure we choose the attribute concept $\mu(\geq 1010)$ for the representation of the notion of "high pressure", and call this formal concept **High**. Combined with the formal concept **Monday** in the time scale we like to form the tuple (**High, Monday**) as our formal representation of such an abstract "object" like "**the High at Monday**". To express formally that an

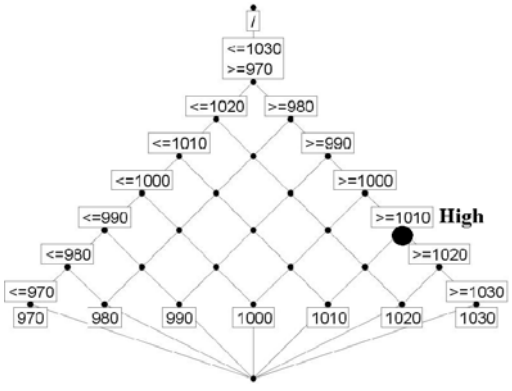


Fig. 2. The semantic scale for *pressure*

abstract object like **High** can move over some landscape we have to connect the information about pressure and time with the places and their longitude and latitude values. For that purpose we use the "artificial key" of the instances. The theory for Temporal Conceptual Semantic Systems will be given in the next sections. As a result of that theory we show a visualization of the movement of the selected high pressure zone in Fig. 3.

Concerning the construction of the weather map in Fig. 3 we just mention the main steps:

- The construction of a grid of latitude-longitude-values as a semi-product of two ordinal scales for latitude and longitude.
- The construction of a map by embedding places using their latitude-longitude-values.
- The embedding of the traces of the chosen tuples of concepts, as for example the traces of the tuple (**High**, **Monday**) and (**High**, **Tuesday**) as visualized by ellipses in Fig. 3.
- The graphical representation of the time relation by arrows connecting the traces of the chosen tuples.

The details of the construction of the weather map in Fig. 3 can be seen in [31].

3 Conceptual Semantic Systems

3.1 Main Ideas

For describing systems like for example physical or technical systems, social systems or the system of a family as represented by a psychological questionnaire we focus on system descriptions by data. We use Conceptual Semantic Systems as a simple and general tool for describing observations in reality. If there are

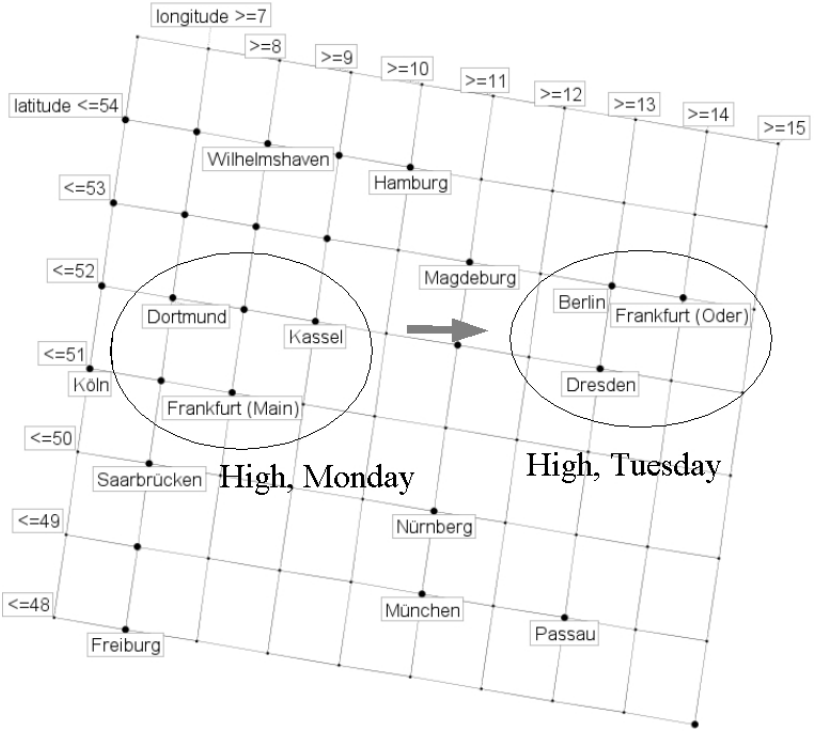


Fig. 3. A Weather Map with a Moving High Pressure Zone

regularities in a system, they can be described by dependencies among many-valued attributes, in the most simple case by functional dependencies.

In the following definition of a Conceptual Semantic System (CSS) we start with a family of formal contexts whose formal concepts are used as "basic semantic concepts" for the description of statements about some part of the reality. These statements are formally represented in the rows of a data table whose values are the chosen basic semantic concepts.

Remark: In the definition of a Conceptual Semantic System we do not explicitly represent the relational aspect of a statement as it is done in Conceptual Graphs [16, 17], in Power Context Families and Concept Graphs [19]. Recently, these relational structures have been combined with temporal CSSs by the author in Temporal Relational Semantic Systems [33, 34, 35].

In the data table of a CSS the rows are indexed by a set G which plays the role of an "artificial" key of the data table. G is also the set of formal objects of the semantically derived context which will be defined later. Our advice, that these formal objects should not be used in applications for the representation of objects in reality, yields the necessity of an alternative representation of objects in Conceptual Semantic Systems which will be given in the following sections.

3.2 Definition of a Conceptual Semantic System

Definition 1. ***"Conceptual Semantic System"***
Let M be a set, and for each $m \in M$ let $\mathbb{S}_m := (G_m, N_m, I_m)$ be a formal context and $\underline{\mathfrak{B}}(\mathbb{S}_m)$ its concept lattice. Let G be a set and

$$\lambda : G \times M \to \bigcup_{m \in M} \mathfrak{B}(\mathbb{S}_m)$$

be a mapping such that $\lambda(g, m) \in \mathfrak{B}(\mathbb{S}_m)$.
Then the quadruple

$$\mathfrak{K} := (G,\ M,\ (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m \in M}, \lambda)$$

is called a Conceptual Semantic System (CSS) with semantic scales $(\mathbb{S}_m)_{m \in M}$. The elements of M are called many-valued attributes; the elements of G are called instances. We write $m(g) := \lambda(g, m)$ and $m(G) := \{m(g) \mid g \in G\}$.

We mention that the mapping λ can be interpreted as a (possibly infinite) data table whose values $m(g)$ "in column m" are formal concepts of the given semantic scale $\mathbb{S}_m$. For any instance g the tuple $(m(g)|m \in M)$ is interpreted as a short description of a statement connecting the concepts $m(g)$ where $m \in M$. A concept $m(g)$ may denote for example the grammatical subject, or the grammatical object, or the grammatical predicate of a statement. That allows for the representation of arbitrary, not only binary, relations; that has been made explicit in [35].

One of the main points in our intended interpretation of Conceptual Semantic Systems is that the instances are not interpreted as meaningful concepts, as for example objects like persons or particles. That differs strongly from the usual interpretation of many-valued contexts [10] where the formal objects are intended to represent objects in "reality". As formal objects they have to form a key in the data table of the many-valued context. In Conceptual Semantic Systems we are much more flexible in practical applications since we do not need an object domain whose objects form a key in the data table. Now we represent the information given in the 'data table' λ of a CSS by a single formal context, called the *semantically derived context* of the CSS.

Definition 2. ***"Semantically derived context"***
Let $\mathfrak{K} := (G,\ M,\ (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m \in M}, \lambda)$ be a CSS with semantic scales

$$\mathbb{S}_m = (G_m, N_m, I_m)\ (m \in M),$$

*and let int(**c**) denote the intent of a concept **c**. Then the formal context*

$$\mathbb{K} := (G, N, J) \text{ where } N := \{(m, n) \mid m \in M, n \in N_m\} \text{ and}$$
$$gJ(m, n) :\Longleftrightarrow n \in int(m(g))$$

is called the semantically derived context of $\mathfrak{K}$.

We mention that a CSS $\mathfrak{K} :=$ (G, M, $(\underline{\mathfrak{B}}(\mathbb{S}_m))_{m \in M}, \lambda$) can be reconstructed from its semantically derived context $\mathbb{K} := (G, N, J)$ and the semantic scales $(\mathbb{S}_m)_{m \in M}$, since each value $(A, B) = \lambda(g, m)$ is uniquely determined by the

intent of the object concept of g in the $m-$part $\mathbb{K}_m := (G, \{m\} \times N_m, J \cap (G \times (\{m\} \times N_m)))$ of $\mathbb{K}$. The corresponding "reconstruction property" for scaled many-valued contexts does not hold if there exists at least one scale $\mathbb{S}_m$ whose object concept mapping γ_m is not injective. The precise connection between the semantically derived context of a CSS and the derived context of a many-valued context is described in [31].

3.3 Object Representation by Tuples of Semantic Concepts

In the main interpretation of contexts and many-valued contexts the formal objects are used to represent certain "objects" in reality, like for example "living beings" in the formal context of Figure 1.1 in [10], p. 18. That led to difficulties in the formal representation of temporal systems where time granules, like for example hours or days, are used as formal objects. Clearly, in such data we do not want to represent other kinds of objects, like persons, also as formal objects. A very successful formal representation was the notion of an *actual object*, defined as a pair (p, t) of an object (for example a person) and a time granule; these actual objects had been chosen as formal objects of such "temporal" contexts. That led the author to the notion of a CTSOT (Conceptual Time Systems with actual Objects and a Time relation) [22, 23, 27] which are very well suited for the representation of life tracks of objects. The disadvantage of the CTSOTs is that *distributed objects* like a high pressure zone on a weather map can not be represented in a CTSOT since each actual object has, as a formal object, exactly one object concept in each part of the derived context - and is therefore not distributed.

That led the author to the notion of a CSS with the interpretation that the usually numerous kinds of general objects, like for example persons and places, should be treated in a symmetric way in the CSS. Hence none of these kinds of objects should play the special role of being represented by the formal objects. Therefore, we interpret the set of formal objects of a CSS as a "meaningless syntactical key" which can be understood as a set of labels for the rows of a data table. It is very advantageous in practical applications that we do not have to search for a kind of objects in practice which can be represented as a key of our intended data table.

In the following we represent objects, like for example persons, days or towns, as formal concepts of the chosen semantic scales of a CSS. Then we would also like to have a formal representation of concatenations of formal concepts of different semantic scales as for example the notions of actual objects like (**High**, **Monday**). For that purpose we introduce for a given CSS the set of tuples of semantic concepts over M.

Definition 3. ***"Tuples of semantic concepts"***
Let $\mathfrak{K} := (G, M, (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m \in M}, \lambda)$ *be a CSS. Then*

$$\mathcal{T}(M) := \{(\boldsymbol{c}_m)_{m \in M^*} \mid \boldsymbol{c}_m \in \mathfrak{B}(\mathbb{S}_m), \emptyset \neq M^* \subseteq M\}$$

is called the set of tuples of semantic concepts over M.

Remark: Of course a tuple $(\mathbf{c}_m)_{m\in M^*}$ is understood as a mapping which maps each element $m \in M^*$ to a formal concept $\mathbf{c}_m \in \mathfrak{B}(\mathbb{S}_m)$.

Definition 4. ***"Key of a CSS"***
Let $\mathfrak{K} := (G, M, (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m\in M}, \lambda)$ be a CSS. A set $K \subseteq M$ is called a key of $\mathfrak{K}$, if the mapping $\lambda_K : G \longrightarrow \mathcal{T}(K)$ where $\lambda_K(g) := (\lambda(g, m))_{m\in K}$ is injective.

In the following we do not assume that there exists a key of a CSS $\mathfrak{K}$. Even the attribute set M need not be a key, that is "multiple rows" are not forbidden. Clearly, if there exists a key K of $\mathfrak{K}$, then the tuples of $\lambda_K(G)$, called the occurring K-tuples, could be used for the representation of "objects in reality" which could then be represented by the formal objects in G. That corresponds to the standard interpretation of formal objects in contexts and many-valued contexts.

The tuples of semantic concepts will be used in the following twice: first, we will use it in the next section in the definition of a *selection*. Second, we will introduce a specified subset $\mathcal{O}$ of $\mathcal{T}$(M) as the set of temporal objects in the definition of a Temporal Conceptual Semantic System. In our example of a moving high pressure zone the 1-tuple (**High**) is a temporal object. To define for a temporal object its *state at a time granule* in some space, for example in a weather map, as indicated in Fig. 3 by an ellipse with the label (**High**, **Monday**), we introduce in the next section the notion of a σ-Q-trace of a tuple where σ is a *selection* and Q is a *view*.

3.4 Views, Selections, and Traces

Now we recall the definitions of *views*, *selections*, and *traces* from [32].

Definition 5. ***"View"***
Let $\mathbb{K} := (G, N, J)$ be a formal context. Then any subset $Q \subseteq N$ is called a ***view*** *of $\mathbb{K}$. The subcontext $\mathbb{K}_Q := (G, Q, J \cap (G \times Q))$ is called the $Q-$part of $\mathbb{K}$.*

In the following the concept lattice of the $Q-$part of a view Q of the semantically derived context of a CSS will be used as a "landscape" into which further information is embedded. To describe this embedding of information we use the following notion of a *selection*.

Definition 6. ***"Instance selection"***
Let $\mathfrak{K} := (G, M, (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m\in M}, \lambda)$ be a CSS, $\mathcal{T}(M)$ the set of tuples of semantic concepts over M. For a subset $\mathcal{T} \subseteq \mathcal{T}(M)$ any mapping

$$\sigma : \mathcal{T} \to \mathfrak{P}(G) := \{X \mid X \subseteq G\}$$

is called an instance selection on $\mathcal{T}$. For $\boldsymbol{c} \in \mathcal{T}$ the set $\sigma(\boldsymbol{c}) \subseteq G$ is called the instance selection of $\boldsymbol{c}$ or the selection of $\boldsymbol{c}$.

The standard instance selection in database theory is

$$\sigma_{db}((\mathbf{c}_m)_{m\in M^*}) := \{g \in G \mid \forall_{m\in M^*}\ m(g) = \mathbf{c}_m\}.$$

In Conceptual Semantic Systems we may use other instance selections, for example

$$\sigma_{\alpha}((\mathbf{c}_m)_{m\in M^*}) := \{g \in G \mid \forall_{m\in M^*}\ m(g) \leq \mathbf{c}_m\}.$$

The instance selection σ_α is used in our example to select for the formal concept **High** of the pressure scale all instances g such that the pressure value of g is a sub-concept of **High**. It is noteworthy that **High** does not occur as a pressure value in Tab. 1.

Definition 7. ***"σ-Q-trace of a tuple"***
Let $\mathfrak{K} := (G, M, (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m\in M}, \lambda)$ be a CSS with semantically derived context $\mathbb{K} := (G, N, J)$, σ an instance selection on $\mathcal{T} \subseteq \mathcal{T}(M)$, $\boldsymbol{c} \in \mathcal{T}$, and $Q \subseteq N$ a view with object concept mapping γ_Q. Then the set $\gamma_Q(\sigma(\boldsymbol{c}))$ is called the σ-Q-trace of the tuple $\boldsymbol{c}$.

In our example of the moving high pressure zone the selection of the tuple (**High**, **Monday**) is chosen as the set of all instances g such that $pressure(g) \geq 1010$ and $time(g) =$ **Monday**. The σ-Q-trace of this tuple is marked in Fig. 3 by the left ellipse. That is conceptually the same technique as to represent towns, mountains, and rivers in a geographical map. In that sense, the notion of a trace generalizes also the geometrical notion of a volume.

3.5 Precise and Distributed Tuples

Definition 8. ***"precise and distributed tuples"***
Let $\mathfrak{K} := (G, M, (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m\in M}, \lambda)$ be a CSS with semantically derived context $\mathbb{K} := (G, N, J)$, σ an instance selection on $\mathcal{T} \subseteq \mathcal{T}(M)$, $\boldsymbol{c} \in \mathcal{T}$, and $Q \subseteq N$ a view with object concept mapping γ_Q. Then $\boldsymbol{c}$ is called

- *σ-precise in $\mathfrak{B}(\mathbb{K}_Q)$* $\iff |\gamma_Q(\sigma(\boldsymbol{c}))| = 1$;
- *σ-distributed in $\mathfrak{B}(\mathbb{K}_Q)$* $\iff |\gamma_Q(\sigma(\boldsymbol{c}))| \geq 2$.

The tuple (**High**, **Monday**) is a good example for a *distributed* tuple since its σ-Q-trace has more than one object concept. Each of the towns in Fig. 3 has a trace with exactly one object concept. They are examples of *precise* 1-tuples.

4 Temporal Conceptual Semantic Systems

Temporal Concept Analysis has been introduced by the author [22, 23] as the theory of temporal phenomena described with tools of Formal Concept Analysis. The notion of a Temporal Conceptual Semantic System generalizes previous notions of temporal systems in Temporal Concept Analysis. It covers discrete, continuous, and hybrid systems as special cases.

From [31, 32] we recall the definition of a Temporal Conceptual Semantic System (TCSS). There are four main ideas which are mathematically described in the definition of a TCSS:

1. A TCSS should be a CSS having a specified many-valued *time attribute* T whose scale contains all temporal concepts which are needed for the given purpose. Instead of a single time attribute we could also introduce a non-empty set of time attributes. The corresponding changes are explicitly written down by the author in his paper [35] on Temporal Relational Semantic Systems.

2. It also should have a specified set $\mathcal{O}$ of *temporal objects* which are used to describe moving objects (like cars) as opposed to static objects (like houses), clearly with respect to the given purpose.
3. Each temporal object $\mathfrak{o} \in \mathcal{O}$ is associated with a binary relation $\mathcal{R}_\mathfrak{o}$ of *base transitions* where each base transition is a pair of formal concepts from the time scale $\mathbb{S}_T$. Each base transition of the temporal object $\mathfrak{o}$ describes one step of $\mathfrak{o}$ in time; other temporal objects may do other steps in time.
4. Each temporal object $\mathfrak{o}$ of a TCSS is represented as a tuple of semantic concepts over M:
$\mathfrak{o} := (\mathbf{c}_m)_{m \in M^*} \in \mathcal{T}(\mathrm{M})$.

Definition 9. ***"Temporal Conceptual Semantic System"***
Let $\mathfrak{K} := (G,\ M,\ (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m \in M}, \lambda)$ *be a CSS,* $T \in M$, $\mathcal{O} \subseteq \mathcal{T}(M)$ *, and for each* $\mathfrak{o} \in \mathcal{O}$ *let* $\mathcal{R}_\mathfrak{o}$ *be a binary relation on* $\mathfrak{B}(\mathbb{S}_T)$*. Then the quadruple*

$$(\mathfrak{K}, T, \mathcal{O}, (\mathcal{R}_\mathfrak{o})_{\mathfrak{o} \in \mathcal{O}})$$

is called a Temporal Conceptual Semantic System (TCSS) with time attribute T*, the set* $\mathcal{O}$ *of temporal objects, and for each temporal object* $\mathfrak{o} \in \mathcal{O}$ *its time relation* $\mathcal{R}_\mathfrak{o}$*. The elements of* $\mathcal{R}_\mathfrak{o}$ *are called base transitions of* $\mathfrak{o}$*. The elements of* $\mathfrak{B}(\mathbb{S}_T)$ *are called time granules.*

One of the central notions in temporal systems is that of a state which is introduced in the following section.

4.1 States

The notion of a state is used in all system descriptions, but, to the best of my knowledge, nowhere defined mathematically clear in such a way that it is connected with a formal description of objects, time, and granularity [27, 36]. That seems to be necessary if we wish to say that "an object is at some time granule in some state". The author has introduced in [20] a definition of a state in a Conceptual Time System as an object concept of a formal object which was interpreted as a time granule. Later on this definition was extended to CTSOTs where pairs (p, t) of objects (e.g persons) and time granules are used as formal objects. That led to a very useful notion of a state of an object at some time granule, but that state was also just a single object concept. That did not fit with the idea that a state should be connected with a certain (probability) distribution, as for example in Quantum Theory [2, 15]. The introduction of CSSs and its distributed objects yields a solution also for this case [28].

The following definition generalizes the notion of a state as defined in [28] by introducing arbitrary instance selections and the object representation by tuples of semantic concepts.

Definition 10. ***"State of a temporal object at a time granule"***
Let $(\mathfrak{K}, T, \mathcal{O}, (\mathcal{R}_\mathfrak{o})_{\mathfrak{o} \in \mathcal{O}})$ *be a TCSS and* $\mathfrak{K} = (G,\ M,\ (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m \in M}, \lambda)$*. Let* σ *be an instance selection and* Q *be a view of the semantically derived context* $\mathbb{K} = (G, N, J)$ *of* $\mathfrak{K}$*. For each temporal object* $\mathfrak{o} = (\boldsymbol{c}_m)_{m \in M^*} \in \mathcal{O}$ *where* $T \notin M^*$

and each time granule $\boldsymbol{t} \in \mathfrak{B}(\mathbb{S}_T)$ *the tuple* $(\mathfrak{o}, \boldsymbol{t})$ *is called an "actual object". The* σ*-*Q*-state of the temporal object* $\mathfrak{o}$ *at time granule* $\boldsymbol{t}$ *is defined as*

$$\gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{t}))$$

which is the σ*-*Q*-trace of the actual object* $(\mathfrak{o}, \boldsymbol{t})$.

Remark: The actual object $(\mathfrak{o}, \mathbf{t})$ is an $(n+1)$-tuple, if the temporal object $\mathfrak{o}$ is an n-tuple of scale concepts.

Examples:

1. If a TCSS $(\mathfrak{K}, T, \mathcal{O}, (\mathcal{R}_\mathfrak{o})_{\mathfrak{o}\in\mathcal{O}})$ has a many-valued attribute $P \neq T$ whose semantic scale is interpreted as a scale for persons, and $\{P, T\}$ forms a key in $\mathfrak{K}$, then the TCSS can be represented as a CTSOT [23, 27], and for each view Q of the semantically derived context $\mathbb{K}$ each actual person $(\mathbf{p}, \mathbf{t}) \in \mathfrak{B}(\mathbb{S}_P) \times \mathfrak{B}(\mathbb{S}_T)$ is precise in $\mathfrak{B}(\mathbb{K}_Q)$ with respect to the usual database selection σ_{db}. In that sense each person is at each time granule at exactly one "place" (= object concept) in $\mathfrak{B}(\mathbb{K}_Q)$. That led for CTSOTs to the notion of a *life track* of a person [23, 27]. The following definitions of *life spaces* and *life tracks* generalize that notion to TCSSs.
2. Distributed states: If the σ-Q-state of the temporal object $\mathfrak{o}$ at time granule $\mathbf{t}$ is distributed in the sense that the actual object $(\mathfrak{o}, \mathbf{t})$ is σ-distributed in $\mathfrak{B}(\mathbb{K}_Q)$, then the relative frequency distribution of the instances $g \in G$ on the object concepts of the σ-Q-state $\gamma_Q(\sigma(\mathfrak{o}, \mathbf{t}))$ forms a probability distribution associated with that state. As opposed to the probability distribution of a quantum mechanical state the probability distribution of a σ-Q-state has a clear frequency interpretation which is relevant for practical measurements. For the special case of the instance selection σ_α and a spatio-temporal CSS this has been discussed in detail by the author in [28].

4.2 Life Space and Life Track

In this section we generalize the notion of life tracks as introduced in the framework of CTSOTs [27] to TCSSs.

Definition 11. ***"Life space and life track of a temporal object"***
Let $(\mathfrak{K}, T, \mathcal{O}, (\mathcal{R}_\mathfrak{o})_{\mathfrak{o}\in\mathcal{O}})$ *be a TCSS and* $\mathfrak{K} = (G, M, (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m\in M}, \lambda)$. *Let* σ *be an instance selection and* Q *be a view of the semantically derived context* $\mathbb{K} = (G, N, J)$ *of* $\mathfrak{K}$. *For each temporal object* $\mathfrak{o} = (\boldsymbol{c}_m)_{m\in M^*} \in \mathcal{O}$ *where* $T \notin M^*$ *we call the set*

$$\mathcal{S}_{\sigma Q}(\mathfrak{o}) := \bigcup_{\boldsymbol{t}\in T(G)} \gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{t}))$$

the σ*-*Q*-life space of the temporal object* o.
The set

$$\mathcal{L}_{\sigma Q}(\mathfrak{o}) := \{((\mathfrak{o}, \boldsymbol{t}), \gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{t}))) \mid \boldsymbol{t} \in T(G)\}$$

is called the labeled σ*-*Q*-life space of* $\mathfrak{o}$.
If $|\gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{t}))| \leq 1$ *for each* $\boldsymbol{t} \in T(G)$, *then we call* $\mathcal{L}_{\sigma Q}(\mathfrak{o})$ *the* σ*-*Q*-life track of* $\mathfrak{o}$.

Remarks:

(1) The notion of a life space of an object can be used for example for the formal representation of the habitat of a biological species from temporal occurrence data of individual animals of that species. In that case it would be useful to choose a semantic scale for the observed animals of that species where the single animals are sub-concepts of a species-concept. For that scale the instance selection σ_α will be useful for the generation of the habitat of that species in some geographic map $\mathfrak{B}(\mathbb{K}_Q)$.
(2) In our example of a moving high pressure zone (cf. Fig. 3) the labeled life space of the concept **High** is visualized by the two labeled ellipses.
(3) If the life space of a temporal object is a life track, then we get the usual labelling of a life track where each state is labeled by all time granules at which the object has "visited" that state.

4.3 Transitions

The basic idea of a transition of an object is a "change of the object from one state to another", for example the *transition of a person from one place to another*. As opposed to automata theory [1, 9] - where transitions are defined as pairs of states, and states are not explicitly connected with time, and objects are not made explicit - we use for the definition of a transition not only a pair of states, but also an object, which performs the transition, and its time relation.

To define a transition in a TCSS $(\mathfrak{K}, T, \mathcal{O}, (\mathcal{R}_\mathfrak{o})_{\mathfrak{o}\in\mathcal{O}})$ we use a temporal object $\mathfrak{o} \in \mathcal{O}$ as a representation of an object which performs the transition. To represent the direction from the initial place of the transition of $\mathfrak{o}$ to the final place of the transition we use a *base transition* $(\mathbf{s}, \mathbf{t}) \in \mathcal{R}_\mathfrak{o}$. To represent the space in which the transition is performed we use the concept lattice $\mathfrak{B}(\mathbb{K}_Q)$ of a suitable view Q of the semantically derived context of $\mathfrak{K}$. To represent the place of an object $\mathfrak{o}$ at a time granule $\mathbf{t}$ in the "map" $\mathfrak{B}(\mathbb{K}_Q)$ we use the σ-Q-trace of the actual object $(\mathfrak{o}, \mathbf{t})$ with respect to some instance selection σ. The following definition of a σ-Q-transition of a temporal object generalizes the corresponding definition of a transition in a CTSOT [27].

Definition 12. ***"σ-Q-transition of a temporal object"***
Let $(\mathfrak{K}, T, \mathcal{O}, (\mathcal{R}_\mathfrak{o})_{\mathfrak{o}\in\mathcal{O}})$ *be a TCSS and* $\mathfrak{K} = (G,\ M,\ (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m\in M}, \lambda)$. *Let* σ *be an instance selection and* Q *be a view of the semantically derived context* $\mathbb{K} = (G, N, J)$ *of* $\mathfrak{K}$. *For each temporal object* $\mathfrak{o} = (\boldsymbol{c}_m)_{m\in M^*} \in \mathcal{O}$ *where* $T \notin M^*$ *and each base transition* $(\boldsymbol{s}, \boldsymbol{t}) \in \mathcal{R}_\mathfrak{o}$ *we call the pair*

$$(((\mathfrak{o}, \boldsymbol{s}), (\mathfrak{o}, \boldsymbol{t})), (\gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{s})), \gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{t}))))$$

the σ-Q-transition of $\mathfrak{o}$ *induced by the base transition* $(\boldsymbol{s}, \boldsymbol{t})$ *leading from the initial place* $((\mathfrak{o}, \boldsymbol{s}), \gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{s})))$ *to the final place* $((\mathfrak{o}, \boldsymbol{t}), \gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{t})))$.

This definition of a transition has the advantage that the initial place as well as the final place of a transition are elements of the labeled life space of the given object.

4.4 The Life Space Digraph

On the labeled life space of an object $\mathfrak{o}$ we can easily introduce a digraph by "transporting" the time relation $\mathcal{R}_\mathfrak{o}$ into the life space. That generalizes the life track digraph for CTSOTs [27].

Definition 13. ***"Life space digraph of a temporal object"***
Let $(\mathfrak{K}, T, \mathcal{O}, (\mathcal{R}_\mathfrak{o})_{\mathfrak{o}\in\mathcal{O}})$ be a TCSS and $\mathfrak{K} = (G,\ M,\ (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m\in M}, \lambda)$. Let σ be an instance selection and Q be a view of the semantically derived context $\mathbb{K} = (G, N, J)$ of $\mathfrak{K}$. For each temporal object $\mathfrak{o} = (\boldsymbol{c}_m)_{m\in M^*} \in \mathcal{O}$ where $T \notin M^*$ the life space digraph of $\mathfrak{o}$ is defined as the directed graph

$$(\mathcal{L}_{\sigma Q}(\mathfrak{o}),\ \hat{\mathcal{R}}_\mathfrak{o})$$

where

$$((\mathfrak{o}, \boldsymbol{s}), \gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{s})))\ \ \hat{\mathcal{R}}_\mathfrak{o}\ \ ((\mathfrak{o}, \boldsymbol{t}), \gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{t}))) \Leftrightarrow \boldsymbol{s}\ \ \mathcal{R}_\mathfrak{o}\ \ \boldsymbol{t}.$$

An example of a small life space digraph with two vertices and one arc can be seen in Fig. 3 where the thick arrow represents the σ-Q-transition of the object **High** induced by the base transition (**Monday**,**Tuesday**).

The life space digraphs can be used very effectively for animations of TCSSs. First computer versions for these animations exist in the computer program SIENA which is a part of the program TOSCANAJ [3, 4].

4.5 Particles and Waves in Temporal Conceptual Semantic Systems

In [26] the author has introduced the notions of particles and waves in spatio-temporal Conceptual Semantic Systems based on the special instance selection σ_α and on a more restrictive notion for objects. In the following definition we generalize that to Temporal Conceptual Semantic Systems with its representation of objects by tuples of semantic concepts and with arbitrary instance selections.

Definition 14. ***"Particles and Waves"***
Let $(\mathfrak{K}, T, \mathcal{O}, (\mathcal{R}_\mathfrak{o})_{\mathfrak{o}\in\mathcal{O}})$ be a TCSS and $\mathfrak{K} = (G,\ M,\ (\underline{\mathfrak{B}}(\mathbb{S}_m))_{m\in M}, \lambda)$. Let σ be an instance selection and Q be a view of the semantically derived context $\mathbb{K} = (G, N, J)$ of $\mathfrak{K}$. A temporal object $\mathfrak{o}$ is called a

- *σ-Q-particle* $\Leftrightarrow |\gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{t}))| \leq 1$ *for all* $\boldsymbol{t} \in T(G)$;
- *σ-Q-wave* $\Leftrightarrow |\gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{t}))| \geq 2$ *for all* $\boldsymbol{t} \in T(G)$;
- *full σ-Q-wave* $\Leftrightarrow \mathfrak{o}$ *is a σ-Q-wave and* $\gamma_Q(\sigma(\mathfrak{o}, \boldsymbol{t})) = \gamma_Q(G)$ *for all* $\boldsymbol{t} \in T(G)$.

It has been proven by the author in [26] that each classical particle in physics can be described by a particle as defined above, and each classical wave can be described by a wave as defined above. Of course many more specific wave notions can be defined.

5 An Application of TCSS: The Behavior of a Distillation Column

5.1 The Data of the Distillation Column

The following application of Temporal Conceptual Semantic Systems in the chemical industry shows the behavior of a distillation column during 20 days. At each day the values of 13 many-valued attributes ("variables") like reflux, energy1, input, and pressure have been measured once. The resulting data table with 20 rows and 13 columns has the variable *day* as its key. This data table is one of the most simple types of a TCSS. There is only one temporal object, namely the distillation column. The time is represented just by the labels from 1 to 20 for the 20 days of measurement. The time relation for the single temporal object is just the predecessor relation on the integers from 1 to 20. All many-valued attributes have been scaled by an ordinal scale having a small chain as its concept lattice.

Even though the data table is quite small it is not easy to understand the behaviour of the distillation column in all 13 variables simultaneously. Therefore, we first studied the variables which are most important for the experts. It was easy for these experts to understand and interpret the behaviour of the distillation column in transition diagrams with two or three variables, like in Fig.4; but they had difficulties understanding not-nested line diagrams representing four or more variables.

5.2 Visualization of a Life Track in a Nested Line Diagram

In the following we visualize the behaviour of the distillation column with respect to the four variables previously mentioned. Preparing this 4-dimensional representation we first show in Fig.4 the life track of the distillation column in a transition diagram for the variables *reflux* and *energy1*, each scaled with a 3-chain where the top concept represents the big values, the bottom concept the small values as indicated by the names of the scale variables, as for example "reflux $\leq$ 133".

We see in Fig.4 that the distillation column is at the first day in the state $\gamma(1)$, the object concept of day 1, labeled by 1. There the *reflux* is small and *energy1* lies in the middle category in this scaling. At the second day the distillation column is in the state $\gamma(2)$, the top concept in this lattice, where each of the two variables has a big value in this scaling. At the third day the destillation column "visits" the state of the first day again. The task to follow the life track visually can be supported by an animation using the computer program SIENA in TOSCANAJ [3, 4].

To visualize now the behavior of the destillation column for the four selected variables we first generate a concept lattice for the two variables *input* and *pressure* where the *input* is scaled in a 5-chain, while the *pressure* is scaled in a 4-chain in the same way as before. Using Peter Becker's latest (unpublished) 2010-version of SIENA one can generate from two given scales a nested line

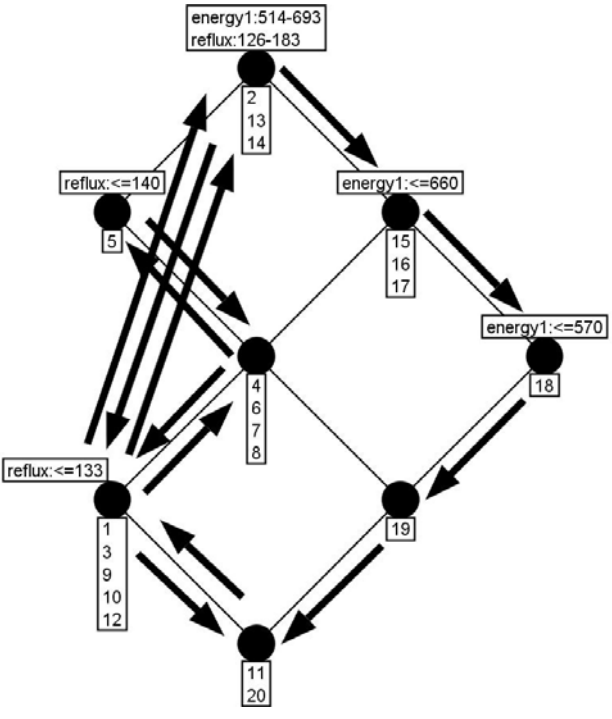


Fig. 4. Life track of a distillation column in a line diagram representing reflux and energy1

diagram. Employing the Temporal Concept Analysis tool of SIENA one can embed life tracks into the nested line diagram. This led to the nested transition diagram in Fig.5 which is explained now.

The coarse structure represented by the 8 ellipses shows the concept lattice for the variables *reflux* and *energy1* as in Fig.4. The fine structure inside the ellipses shows the concept lattice for the variables *input* and *pressure*. Only in the top ellipse the attributes of the fine structure are named. In Fig.5 we see in the coarse structure, that at the first (and the ninth) day the measured value for the *reflux* is ≤ 133, for the *energy1* ≤ 660 and not ≤ 570 as we can see also from Fig.4. Looking at the fine structure in Fig.5 we see that at the first day the values for *input* are ≤ 630 and not ≤ 615 while the first-day values for the *pressure* are ≤ 120 and not ≤ 115. That shows the successful visualization of four variables in each single state of the distillation column. The meaning of the transitions between states is also successfully visualized: as a simple example we just mention, that arrows within a single ellipse represent transitions where the attributes of the coarser structure are constant in the granularity of the outer scale; for example the transition from the state at day 16 to the state at day 17, shortly described by the base transition (16,17), is constant in the outer scale, and changes only the value of the *input* from the top level to the next lower level. Another extrem

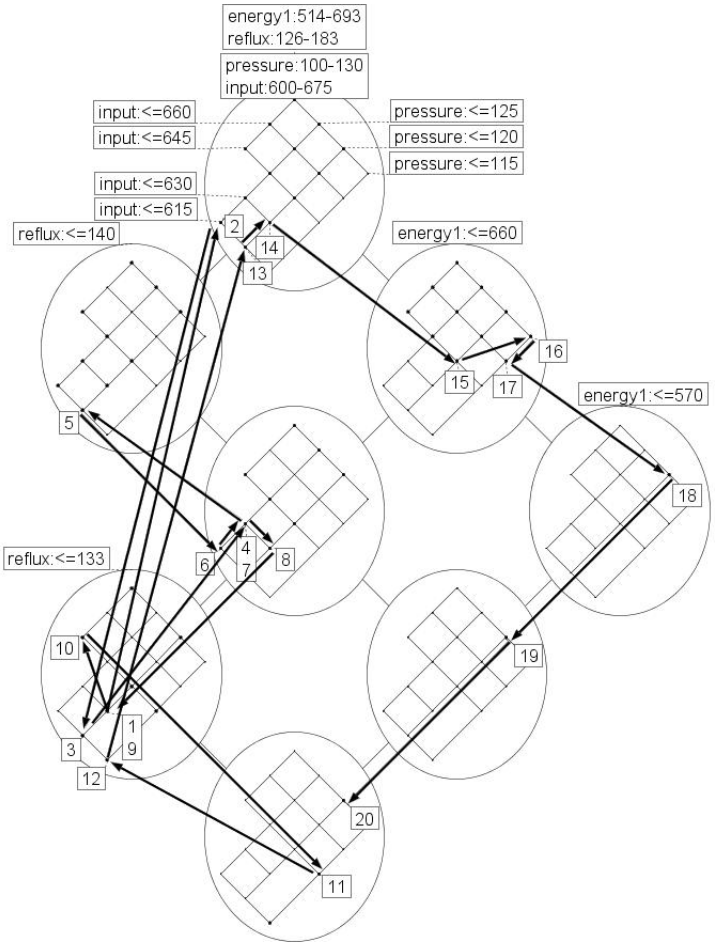


Fig. 5. Life track of a distillation column in a nested line diagram representing reflux, energy1, input and pressure

example is the transition (1,2) which changes all four variables. In the same way the meaning of the other arrows can be easily understood from Fig.5.

The main advantage of the transition diagrams in this application was that the experts of this distillation column realized that they had some wrong ideas about the behaviour of their distillation column. Using transition diagrams in a self-chosen granularity the experts developed visually supported state spaces for the understanding and the control of the production processes in the distillation column.

For another useful application of TCSSs in a biomedical study of disease processes in arthritic patients the reader is referred to the next article in this volume: *Conceptual Representation of Gene Expression Processes* by Johannes Wollbold, René Huber, Raimund Kinne, and Karl Erich Wolff.

6 Conclusions and Future Research

We have generalized the basic notions in Temporal Concept Analysis, namely states, transitions, life tracks, the digraph of a life track, particles and waves to the actual notion of a Temporal Conceptual Semantic System. The main new tools are the representation of objects by tuples of concepts of the semantic scales and the introduction of an instance selection which generalizes the usual instance selection for data bases. As opposed to classical time representations we did not use any linear ordering of the set of time granules, but all the classical time representations as continuous time, discrete time, and the usual time representation with several time attributes for years, months, and days with its cyclic time structures are included as special cases. One of the most promising advantages of Temporal Conceptual Semantic Systems is that the applicability is now much broader than before, since it is no longer necessary to search for an object domain which can be used as a key for the data. The main advantage with respect to the visualization is that trace diagrams combine the ordinal structure of concept lattice with the benefits of Venn diagrams.

Future research will focus on applications in many fields, mainly in the representation of relational data. A first success was the combination of conceptual scaling and concept graphs in Conceptual Relational Semantic Systems [33, 34] and its extension to Temporal Relational Semantic Systems [35]. For practical applications it will be very useful to introduce new graphical tools for the construction of animated trace diagrams into the computer programs of Formal Concept Analysis.

References

[1] Arbib, M.A.: Theory of Abstract Automata. Prentice Hall, Englewood Cliffs (1970)
[2] Auletta, G.: Foundations and Interpretations of Quantum Mechanics. World Scientific Publishing Co. Pte. Ltd., Singapore (2000)
[3] Becker, P.: Multi-dimensional Representation of Conceptual Hierarchies. In: Stumme, G., Mineau, G. (eds.) Proceedings of the 9th International Conference on Conceptual Structures, Supplementary Proceedings, pp. 33–46. Department of Computer Science, University Laval (2001)
[4] Becker, P., Hereth Correia, J.: The ToscanaJ Suite for Implementing Conceptual Information Systems. In: Ganter, B., Stumme, G., Wille, R. (eds.) FCA 2005. LNCS (LNAI), vol. 3626, pp. 324–348. Springer, Heidelberg (2005)
[5] Bertalanffy, L.v.: General System Theory. George Braziller, New York (1969)
[6] Butterfield, J. (ed.): The Arguments of Time. Oxford University Press, Oxford (1999)
[7] Butterfield, J., Isham, C.J.: On the Emergence of Time in Quantum Gravity. In: Butterfield, J. (ed.) The Arguments of Time. Oxford University Press, Oxford (1999)
[8] Castellani, E. (ed.): Interpreting Bodies: Classical and Quantum Objects in Modern Physics. Princeton University Press, Princeton (1998)
[9] Eilenberg, S.: Automata, Languages, and Machines, vol. A. Academic Press, London (1974)

[10] Ganter, B., Wille, R.: Formal Concept Analysis: mathematical foundations. Springer, Heidelberg (1999); German version: Springer, Heidelberg (1996)
[11] Kalman, R.E., Falb, P.L., Arbib, M.A.: Topics in Mathematical System Theory. McGraw-Hill Book Company, New York (1969)
[12] Lin, Y.: General Systems Theory: A Mathematical Approach. Kluwer Academic/Plenum Publishers, New York (1999)
[13] Mesarovic, M.D., Takahara, Y.: General Systems Theory: Mathematical Foundations. Academic Press, London (1975)
[14] Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer Academic Publishers, Dordrecht (1991)
[15] Neumann, J.v.: Mathematical Foundations of Quantum Mechanics (engl. translation of Neumann, J.v.: Mathematische Grundlagen der Quantenmechanik. Springer, Berlin (1932)). University Press, Princeton (1932)
[16] Sowa, J.F.: Conceptual structures: information processing in mind and machine. Addison-Wesley, Reading (1984)
[17] Sowa, J.F.: Knowledge representation: logical, philosophical, and computational foundations. Brooks Cole Publ. Comp., Pacific Grove (2000)
[18] Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Rival, I. (ed.) Ordered sets, pp. 445–470. Reidel, Dordrecht (1982); Reprinted in: Ferré, S., Rudolph, S. (eds.): Formal Concept Analysis. ICFCA 2009. LNAI 5548, pp. 314–339. Springer, Heidelberg (2009)
[19] Wille, R.: Conceptual Graphs and Formal Concept Analysis. In: Delugach, H.S., Keeler, M.A., Searle, L., Lukose, D., Sowa, J.F. (eds.) ICCS 1997. LNCS (LNAI), vol. 1257, pp. 290–303. Springer, Heidelberg (1997)
[20] Wolff, K.E.: Concepts, States, and Systems. In: Dubois, D.M. (ed.) Proceedings of Computing Anticipatory Systems. American Institute of Physics, Conference, vol. 517, pp. 83–97 (2000)
[21] Wolff, K.E.: A Conceptual View of Knowledge Bases in Rough Set Theory. In: Ziarko, W.P., Yao, Y. (eds.) RSCTC 2000. LNCS (LNAI), vol. 2005, pp. 220–228. Springer, Heidelberg (2001)
[22] Wolff, K.E.: Temporal Concept Analysis. In: Mephu Nguifo, E., et al. (eds.) ICCS 2001 International Workshop on Concept Lattices-Based Theory, Methods and Tools for Knowledge Discovery in Databases, pp. 91–107. Stanford University, Palo Alto (2001)
[23] Wolff, K.E.: Transitions in Conceptual Time Systems. International Journal of Computing Anticipatory Systems 11, 398–412 (2002)
[24] Wolff, K.E.: Interpretation of Automata in Temporal Concept Analysis. In: Priss, U., Corbett, D.R., Angelova, G. (eds.) ICCS 2002. LNCS (LNAI), vol. 2393, pp. 341–353. Springer, Heidelberg (2002)
[25] Wolff, K.E.: Concepts in Fuzzy Scaling Theory: Order and Granularity. Fuzzy Sets and Systems 132, 63–75 (2002)
[26] Wolff, K.E.: 'Particles' and 'Waves' as Understood by Temporal Concept Analysis. In: Wolff, K.E., Pfeiffer, H.D., Delugach, H.S. (eds.) ICCS 2004. LNCS (LNAI), vol. 3127, pp. 126–141. Springer, Heidelberg (2004)
[27] Wolff, K.E.: States, Transitions, and Life Tracks in Temporal Concept Analysis. In: Ganter, B., Stumme, G., Wille, R. (eds.) FCA 2005. LNCS (LNAI), vol. 3626, pp. 127–148. Springer, Heidelberg (2005)
[28] Wolff, K.E.: States of Distributed Objects in Conceptual Semantic Systems. In: Dau, F., Mugnier, M.-L., Stumme, G. (eds.) ICCS 2005. LNCS (LNAI), vol. 3596, pp. 250–266. Springer, Heidelberg (2005)

[29] Wolff, K.E.: A Conceptual Analogue of Heisenberg's Uncertainty Relation. In: Ganter, B., Kwuida, L. (eds.) Contributions to ICFCA 2006, pp. 19–30. Verlag Allgemeine Wissenschaft (2006)
[30] Wolff, K.E.: Conceptual Semantic Systems - Theory and Applications. In: Goncharov, S., Downey, R., Ono, H. (eds.) Mathematical Logic in Asia, pp. 287–300. World Scientific, New Jersey (2006)
[31] Wolff, K.E.: Basic Notions in Temporal Conceptual Semantic Systems. In: Gély, A., Kuznetsov, S.O., Nourine, L., Schmidt, S.E. (eds.) Contributions to ICFCA 2007, pp. 97–120. Clermont-Ferrand, France (2007)
[32] Wolff, K.E.: Applications of Temporal Conceptual Semantic Systems. In: Zagoruiko, N.G., Palchunov, D.E. (eds.) Knowledge - Ontology - Theory, vol. 2, pp. 3–16. Russian Academy of Sciences. Sobolev Institute for Mathematics, Novosibirsk (2007)
[33] Wolff, K.E.: Relational Semantic Systems, Power Context Families, and Concept Graphs. In: Wolff, K.E., Rudolph, S., Ferré, S. (eds.) Contributions to ICFCA 2009, pp. 63–78. Verlag Allgemeine Wissenschaft, Darmstadt (2009)
[34] Wolff, K.E.: Relational Scaling in Relational Semantic Systems. In: Rudolph, S., Dau, F., Kuznetsov, S.O. (eds.) ICCS 2009. LNCS (LNAI), vol. 5662, pp. 307–320. Springer, Heidelberg (2009)
[35] Wolff, K.E.: Temporal Relational Semantic Systems. In: Croitoru, M., Ferré, S., Lukose, D. (eds.) ICCS 2010. LNCS (LNAI), vol. 6208, pp. 165–180. Springer, Heidelberg (2010)
[36] Zadeh, L.A.: The Concept of State in System Theory. In: Mesarovic, M.D. (ed.) Views on General Systems Theory, pp. 39–50. John Wiley & Sons, New York (1964)
[37] Zadeh, L.A.: Fuzzy sets. Information and Control 8, 338–353 (1965)
[38] Zadeh, L.A.: The concept of a linguistic variable and its application to approximate reasoning. Part I: Inf. Science 8, 199–249, Part II: Inf. Science 8, 301–357; Part III: Inf. Science 9, 43–80 (1975)

Conceptual Representation of Gene Expression Processes

Johannes Wollbold[1], René Huber[2,3], Raimund Kinne[2], and Karl Erich Wolff[4]

[1] Steinbeis Transfer Center for Proteome Analysis, Rostock, Germany
jwollbold@gmx.de
[2] University Hospital Jena, Experimental Rheumatology Group
raimund.w.kinne@med.uni-jena.de
[3] Institute of Clinical Chemistry, Hannover Medical School, Hannover, Germany
huber.rene@mh-hannover.de
[4] University of Applied Sciences, Mathematics and Science Faculty, Darmstadt, Germany
karl.erich.wolff@t-online.de

Abstract. The present work visualizes and interprets gene expression data of arthritic patients using the mathematical theory of Formal Concept Analysis (FCA). For the purpose of representing gene expression processes we employ the branch of Temporal Concept Analysis (TCA) which has been introduced during the last ten years in order to support conceptual reasoning about temporal phenomena. In TCA, movements of general objects in abstract or "real" space and time can be described in a conceptual framework. For our purpose in this paper we only need a special case of the general notion of a *Conceptual Semantic System (CSS)*, namely a *Conceptual Time System with actual Objects and a Time relation (CTSOT)*. In the theory of CTSOTs, there are clear mathematical definitions of notions of objects, states, situations, transitions and life tracks. It is very important for our application that these notions are compatible with the granularity of the chosen *scaling* of the original data.

This paper contributes to the biomedical study of disease processes in rheumatoid arthritis (RA) and the inflammatory disease control osteoarthritis (OA), focusing on their molecular regulation. Time series of messenger RNA (mRNA) concentration levels in synovial cells from RA and OA patients were measured for a period of 12 hours after cytokine stimulation. These data are represented simultaneously as life tracks in transition diagrams of concept lattices constructed from the mRNA measurements for small sets of interesting genes. Biologically interesting differences between the two groups of patients are revealed. The transition diagrams are compared to literature and expert knowledge in order to explain the observed transitions by influences of certain proteins on gene transcription and to deduce new hypotheses concerning gene regulation.

1 Introduction

The present work is based on a cooperation among scientists from medicine, biology, and mathematics. In the first part of this introduction, the biomedical side is presented, in the second part the mathematical side, focussing on a

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 79–100, 2011.

knowledge representation using Formal Concept Analysis (FCA) [10] and its branch of Temporal Concept Analysis (TCA) [19,24].

1.1 Gene Expression Processes in Arthritic Patients

In this paper we investigate gene expression time series for patients suffering from rheumatoid arthritis (RA). RA is characterized by chronic inflammation, accompanied by the destruction of multiple joints perpetuated by the synovial membrane (SM, or synovium) (Figure 1). A major component of the inflamed SM are activated, aggressive synovial fibroblasts (SFB, or synoviocytes).

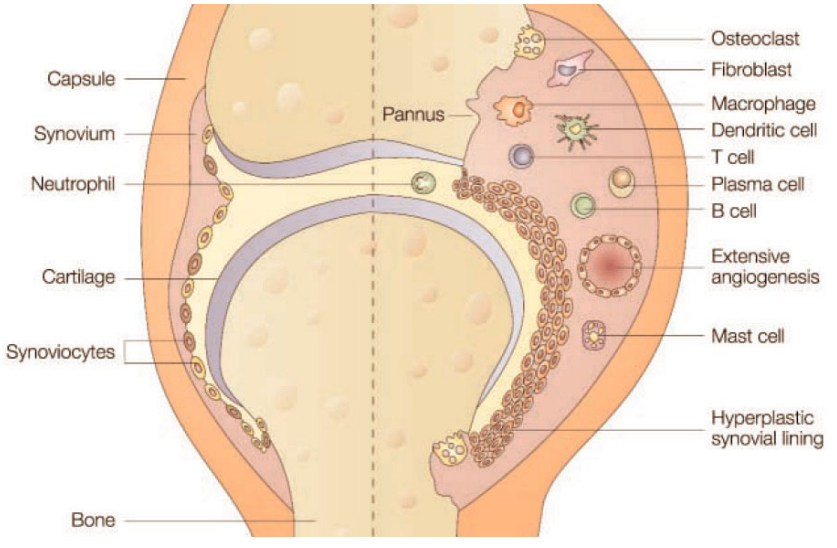


Fig. 1. The knee joint: normal morphology (left) and schematic representation of rheumatoid arthritis features (right) [13, Figure 1]

To understand normal and destructive cellular reactions, the biomedical theory has developed models describing processes at the molecular level. A fundamental process is *gene expression*, which is a sequence of two phases, *transcription* and *translation* (Figure 2): i) during *transcription*, messenger ribonucleic acid (mRNA) is produced from the genetic DNA template (i.e., a DNA sequence coding for a single protein); and ii) during *translation*, proteins are built from amino acids using these mRNA templates. [1]

Proteins are the main regulators of life processes; they activate almost every chemical reaction within an organism as enzymes, build new structures during cell division or transduce biochemical signals from the cell surface to the cytoplasm and the nucleus, e.g., by phosphorylation of target proteins. These signals can activate another class of proteins, the *transcription factors*, which bind to the DNA and are thus able to initiate, enhance or repress transcription. [1] These mutual dependencies are represented as *gene regulatory networks* like in Figure 5.

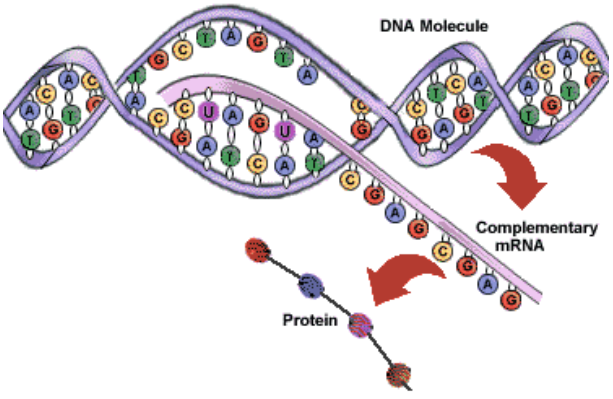


Fig. 2. Gene expression: the two phases of transcription and translation [www.scientificpsychic.com/fitness/aminoacids1.html]

In normal joints, SFB show a balanced expression of proteins, regulating the formation and degradation of the *extracellular matrix* (ECM), which provides structural support to the cells. In RA, however, SFB have a decisive influence on the development and progression of the disease by predominant expression and secretion of pro-inflammatory cytokines and tissue-degrading enzymes, thus maintaining joint inflammation and degradation of ECM components of cartilage and bone [8], [13]. In addition, enhanced formation of soft ECM components (an attempt of wound healing resulting in fibrosis) in the affected joints is also driven by SFB, which express enhanced amounts of collagens.

Central transcription factors involved as key players in the RA pathogenesis and activation of SFB are AP1, NFκB, ETS1, and SMAD [26]. These transcription factors show binding activity for their cognate recognition sites in promoters of inflammation-related cytokines (e.g., TNFα) and matrix-degrading target genes, e.g., collagenase (MMP1) and stromelysin 1 (MMP3). The latter are highly expressed in RA and contribute to tissue degradation by destruction of ECM components.

Based on this knowledge we analyzed the expression of 18 genes, which can be classified into five functional groups:

1. Structural proteins (COL1A1 and COL1A2)
2. Enzymes degrading ECM molecules (MMP1, -3, -9, and -13)
3. Molecules inhibiting these proteases (TIMP1)
4. Transcription factors (ETS1, FOS, JUN, JUNB, JUND, NKFB1, SMAD3, SMAD4, SMAD7) regulating the expression of the above genes
5. External signaling molecules TNFα (TNF) and TGFβ (TGFB1)

Virtually all of these genes can be expressed in fibroblasts and are involved in ECM turnover.

Since several years, progress of experimental techniques in molecular biology has allowed to collect *high-throughput* data for large sets of genes, e.g., by a

single gene chip (*microarray*) measuring the concentration levels of the mRNA for almost all 25.000 human genes. However, conclusions from such data have to be drawn with caution, since mRNA and protein concentrations sometimes are only weakly correlated.

In our approach we investigate the microarray measurements of the mRNA levels for the 18 genes mentioned above.

A main task of systems biology is to model the behavior of biological systems by regulatory network models. A classical approach are *Boolean networks*, first proposed by Kauffman et al. [11]. A Boolean network mainly consists of a directed graph with n nodes, which are interpreted as genes. At each point of time, formally represented as a non-negative integer, a gene may be either active ('*on*'; gene expression 'yes') or inactive ('*off*'; gene expression 'no'). Therefore, a *state* of a Boolean network at a time point t is defined as a sequence $(s_1, ..., s_n)$ of n binary *state values* s_i where $s_i = 0$ (respectively 1) if and only if the i-th gene is off (respectively on) at time point t. The arcs of the Boolean network are used to represent temporal dependencies between genes; in [11] it is assumed that there is an integer $k \leq n$ so that each gene has exactly k incoming arcs from k genes which influence the given gene. This influence of k genes on a gene i is described by a time-independent Boolean function β_i which maps each of the 2^k possible k-tuples of 0's and 1's into the set $\{0, 1\}$. These Boolean functions determine for each state $(s_1, ..., s_n)$ at time t uniquely a state at time $t + 1$. Therefore, the transitions from the state at time t to the state at time $t + 1$ generate a functional digraph; its cycles are interpreted as stable behavioral types of the biological system described by the Boolean network.

In [26], we collected literature knowledge related to the formation and remodelling of ECM, performed dynamic simulations by Boolean networks, compared them to observed time series and made analyses based on the *attribute exploration* algorithm of Formal Concept Analysis (FCA). In the present work we extend this study employing the temporal branch of FCA, Temporal Concept Analysis (TCA), for the representation of gene expression processes. As the main graphical tool we use *transition diagrams* (as for example in Figure 4) for the visualization of these processes. As opposed to Boolean networks in which only a single "system" moves through the state space, the systems in TCA allow for a general and structured representation of multiple temporal objects. In Figure 4 there are six temporal objects, namely patients, each one with its life track. A short introduction into that field will be given in the next sections.

1.2 Temporal Concept Analysis

To give a very short overview over Temporal Concept Analysis (TCA) we mention something about its roots, some basic ideas, and its relation to other temporal theories.

FCA was introduced 1982 by Wille [18], [10]; it is based on order theory, especially lattice theory [6], which has its roots in classical ordinal structures in logic, algebra and geometry. The central definition in FCA is the mathematical notion of a *formal concept* which formalizes the philosophical notion of a *concept*

as a unit of thought with its two parts, the *extension* and the *intension*. Formal concepts are defined in a *formal context* (G, M, I) consisting of a set G of *(formal) objects*, a set M of *(formal) attributes* and a binary relation $I \subseteq G \times M$ between these two sets. If a pair (g, m) of an object g and an attribute m belongs to that relation one says that "g has the attribute m". The set of all formal concepts of a formal context is an ordered set with respect to the conceptual hierarchy (see Chapter 2.2). This ordered set is even a *complete lattice* which means that any set of formal concepts has a *supremum* (the smallest common super-concept) and an *infimum* (the greatest common sub-concept). Concept lattices can be graphically represented by line diagrams which represent the given formal context without any loss of information. Since arbitrary data tables, formally described as *many-valued contexts*, can be represented via *conceptual scaling* as formal contexts, the line diagrams of concept lattices can be used to represent any data table.

The basic idea to represent the notion of a *state* as a formal concept led to the introduction of TCA [19]. Within the framework of *conceptual time systems* it was possible to capture the notion of a *time granule* which generalizes and clarifies the notion of a *time point* or *moment*. The introduction of a general notion for *temporal objects* led to the notion of a *state of a temporal object at a time granule*. To define also *transitions* it was necessary to introduce the notion of a time relation (of a temporal object). That led to the conceptual representation of *life tracks of objects*. The basic structure in which states, transitions and life tracks are defined is called a *Conceptual Time System with actual Objects and a Time relation* (CTSOT) [20,24]. They are graphically represented in *transition diagrams* as in Figure 4, 6, 7 or 9.

To define a life track of an object one needs the property that an object is at exactly one place at each time granule. This property is the basic assumption for the notion of a particle in classical physics. The investigation of this property with respect to the granularity for objects, space and time led to the notion of a temporal *Conceptual Semantic System* (CSS) in which particles and waves can be represented as special *distributed objects* [22]. That improves our conceptual understanding of basic notions in many other temporal theories as for example in classical physics, in quantum physics, in automata theory [21] and in the theory of Turing machines [23]. It also yields a clear mathematical basis for the semantics of situations as discussed in [3,4].

In the following we do not need the general notion of a temporal Conceptual Semantic Systems since the gene expression data can be easily represented as a CTSOT. That will be explained in the next section.

2 Data of Gene Expression Processes and Its Conceptual Representation

2.1 Organization of the Data

The data evaluated in this paper are partly shown in Table 1. The full data table mainly consists of mRNA-measurements applied to cells of six patients and is

published in [26]. Three of these patients (labelled 87, 220, and 221) suffered from rheumatoid arthritis (RA), the other three (190, 202, and 205) suffered from osteoarthritis (OA). Fibroblasts isolated from the joints of these patients have been independently stimulated with two different proteins, namely the pro-inflammatory tumor necrosis factor α (TNFα) and the transforming growth factor β (TGFB1), which acts as a partial antagonist of TNFα. At 0, 1, 2, 4, and 12 hours after stimulation with one of these factors, mRNA measurements for all genes have been taken, using Affymetrix U133 Plus 2.0 chips; in this study, we are interested in the named 18 genes. Logarithmic values (ln-values) of such mRNA-measurements are represented in Table 1.

Table 1. Ln-values of the mRNA measurements for the genes *MMP1, MMP9, and TIMP1*

Key	Time	Patient	Disease	Stimulation	MMP1	MMP9	TIMP1
N87_0	0	N87	RA	TNF	7.91	4.42	10.58
N87_1	1	N87	RA	TNF	9.34	4.02	10.28
N87_2	2	N87	RA	TNF	10.07	4.97	10.63
N87_4	4	N87	RA	TNF	10.42	4.35	10.63
N87_12	12	N87	RA	TNF	10.61	5.18	10.67
G205_0	0	G205	OA	TGFB1	7.08	4.60	10.28
G205_1	1	G205	OA	TGFB1	6.92	4.73	10.44
G205_2	2	G205	OA	TGFB1	6.99	5.02	10.43
G205_4	4	G205	OA	TGFB1	6.98	4.93	10.31
G205_12	12	G205	OA	TGFB1	10.46	5.55	10.51

Reading for example row N87_4 in Table 1, 4 hours after stimulation with TNF, the ln-value of the mRNA measurement for the gene MMP1 in the cells of the RA-patient 87 was 10.42, 4.35 for MMP9, and 10.63 for TIMP1. The row label 'N87_4' represents by 'N' the stimulation with TNF, by '87' the label of the patient, and by '4' the duration of 4 hours after stimulation. Similarly, in 'G205_12' the 'G' represents the stimulation with TGFB1. The full data table has $2 \times 6 \times 5 = 60$ rows and 22 columns, 4 for Time, Patient, Disease and Stimulation, and 18 for genes. Table 1 shows only 10 of the rows and 7 of the columns. The two columns labeled 'Disease' and 'Stimulation' are useful for the conceptual investigation of combinations of the values in these two columns. The conceptual representation and visualization of data will be explained in the following subsection.

2.2 Conceptual Visualization of Data

In the following we represent and visualize our above-mentioned data using FCA. Some basic notions are given for the unexperienced reader, on the basis of the data presented in Table 1. For this purpose, we start with the formal context I_5 shown in Table 2. Its set of formal objects is $G := \{0, 1, 2, 4, 12\}$, the set of time values in Table 1; since we are interested in representing time intervals we choose

the set of formal attributes as $M := \{\leq 0, \leq 1, \leq 2, \leq 4, \leq 12, \geq 0, \geq 1, \geq 2, \geq 4, \geq 12\}$; the incidence relation of I_5 is the set of all those pairs $(g, m) \in G \times M$ which have a cross in Table 2.

Table 2. The formal context I_5 describing temporal intervals

I_5	≤ 0	≤ 1	≤ 2	≤ 4	≤ 12	≥ 0	≥ 1	≥ 2	≥ 4	≥ 12
0	×	×	×	×	×	×				
1		×	×	×	×	×	×			
2			×	×	×	×	×	×		
4				×	×	×	×	×	×	
12					×	×	×	×	×	×

Instead of a mathematical introduction of *formal concepts* and *concept lattices* - which can be found in [10] - we explain the meaning of the main notions using the diagram in Figure 3. This diagram represents the concept lattice of the formal context I_5; we now explain it roughly using some examples.

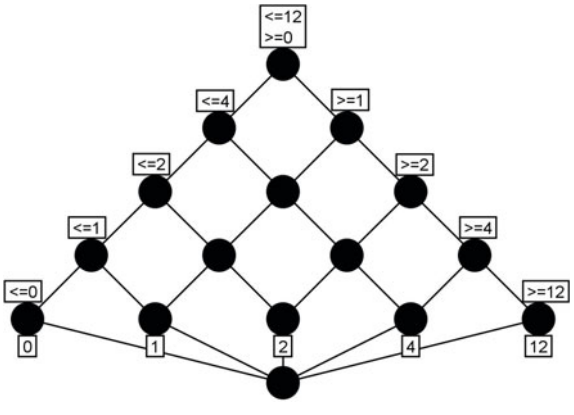


Fig. 3. The concept lattice of the formal context I_5. Circles represent formal concepts and lines the order relation.

First, we mention that all formal objects and all formal attributes of I_5 occur in Figure 3. Each circle describes a formal concept; for example, the circle at the top describes the *top concept*; as any formal concept it consists of a pair (A, B) of two sets, where A is called its *extent*, and B is called its *intent*; for the top concept the extent is the set G of all objects, and the intent is the set of those attributes shared by all objects, hence it is the set $\{<= 12, >= 0\}$. In the cross table of I_5 this top concept can be visualized as the rectangle full of crosses spanned by its extent G and its intent $\{<= 12, >= 0\}$. In the diagram, the extent of any formal concept, represented by some circle, is the set of objects occuring under that circle; its intent is the set of all attributes occuring above this circle. For example, the circle placed vertically under the top concept represents the formal concept $c_{14} := (\{1, 2, 4\}, \{<= 4, <= 12, >= 0, >= 1\})$.

In the following, we need two important special kinds of formal concepts, namely the *object concepts* and the *attribute concepts*. In the diagram they are represented by those circles which are labeled by an object (resp. attribute) name. For example, the circle with the lower label 1 represents the object concept of 1, denoted by $\gamma(1) = (\{1\}, \{<= 1, <= 2, <= 4, <= 12, >= 0, >= 1\})$; the circle with the upper label $[>= 1]$ represents the attribute concept $\mu(>= 1) = (\{1, 2, 4, 12\}, \{>= 1, >= 0, <= 12\})$.

If we are interested in those formal objects which satisfy two attributes, say $\{>= 1, <= 4\}$, then we get the set $\{1, 2, 4\}$ which is the intersection of the extents of the attribute concepts $\mu(>= 1)$ and $\mu(<= 4)$. It can be shown, that the intersection of extents is again an extent; in our example it is the extent of the previously mentioned concept c_{14} which is the infimum of $\mu(>= 1)$ and $\mu(<= 4)$ and the supremum of the object concepts of $1, 2, 4$. This formal concept is neither an object concept nor an attribute concept. In the formal context I_5 each extent of a formal concept is an interval in the set G; as an extreme example, the bottom concept has the empty set as its extent and M as its intent.

For any two formal concepts (A, B), (C, D) of a formal context we say that (A, B) is a subconcept of (C, D) if and only if the extent A is a subset of the extent C:

$$(A, B) \leq (C, D) :\Leftrightarrow A \subseteq C \tag{1}$$

That can be expressed equivalently by the inverse inclusion for the intents: $B \supseteq D$. (A, B) is called a *lower neighbour* of (C, D) if and only if (A, B) is a subconcept of (C, D) and their is no other concept between (A, B) and (C, D); the lower neighbourhood relation is indicated in the diagram by upwards leading lines. Therefore, the diagrams for representing concept lattices are also called *line diagrams*. Using the conceptual hierarchy one can prove (see [10]) that any formal context can be reconstructed from its concept lattice, since an object g has an attribute m if and only if the object concept of g is a subconcept of the attribute concept of m:

$$gIm \Leftrightarrow \gamma(g) \leq \mu(m) \text{ for all } g \in G \text{ and } m \in M. \tag{2}$$

Hence, looking at a concept lattice one can see the whole information given in its formal context. In the next section we explain and use conceptual scaling, the standard method for transforming arbitrary data into a formal context.

2.3 Conceptual Scaling

The main idea for the visualization of arbitrary data is to transform these data into a formal context and to use a well-drawn line diagram of its concept lattice for understanding the data. The main technique for this transformation is conceptual scaling. It has been developed in the Research Group Concept Analysis at Darmstadt University of Technology between 1982 and 1989 (see [9], [10], [18]). In these references, basic definitions for conceptual scaling are introduced: the notions of a *many-valued context* (mathematizing arbitrary data tables), *conceptual scales* (which are formal contexts for the purpose of introducing a

suitable granularity for the values of each many-valued attribute), and the *derived context* of a scaled many-valued context. The derived context represents the many-valued context in the granularity of the chosen scales. The main method for conceptual scaling is *plain scaling* [10]. It is often applied, for example when using thresholds for numerical values of a many-valued attribute. One of the main advantages of conceptual scaling is its flexibility in applications.

As a small example we construct the formal context shown in Table 3 from the MMP1-MMP9-TIMP1 part of Table 1. In this formal context we represent for example the TNF-stimulated patient 87 at time point 0 by $N87\_0$. The incidence relation of this formal context is constructed from Table 1 by a threshold scaling whose main idea is "to make a cross if and only if the value is greater than a suitable threshold".

For these data it is meaningful to introduce for each many-valued attribute two different thresholds depending on the two different stimulations TNF and TGFB1, since the data had been normalized independently. Further, thresholds have to be determined for each gene separately, because the absolute expression values are very different. We have chosen these thresholds as described in Table 4. Since we do not like to mention these thresholds in the names of the attributes of the derived context we call these attributes "MMP1=1", "MMP9=1", "TIMP1=1" to express for example that "MMP1 is on". For the conceptually advanced reader we mention that this scaling is not a plain scaling as described in [10, p. 38].

The stimulation-dependent scaling is explained by the following example: In the column "MMP1=1" of Table 3 the formal object $N87\_1$ has a cross since $N87\_1$ is TNF-stimulated and $\mathrm{MMP1}(N87\_1) = 9.34 > 8.65$; the formal object $G205\_12$ has a cross in the column "MMP1=1" since $G205\_12$ is TGFB1-stimulated and $MMP1(G205\_12) = 10.46 > 7.29$.

These thresholds were determined by the first author using a procedure based upon Wards agglomerative hierarchical clustering method [16]. At each step in this algorithm, which starts from the partition of all singletons, the union of every possible cluster pair is considered and two clusters whose fusion results in minimum *information loss* are combined. Information loss is defined by Ward in terms of an *error sum-of-squares criterion (ESS)*. For our purpose of a coarse threshold scaling we generate for each of the considered genes and for each stimulus a partition with only two clusters for the set of the ln-values of the mRNA measurements. Let this set be $\{x_i | 1 \leq i \leq n\}$ and $x_i \geq x_j$ for $i \leq k$. Then Ward partitioning into two classes signifies determining an index i that minimizes

$$\begin{aligned} ESS :=& \sum_{j=1}^{i}(x_j - \mu_1)^2 + \sum_{j=i+1}^{n}(x_j - \mu_2)^2 \\ =& (i-1)\sigma^2(x_1, ..., x_i) + (n-i-1)\sigma^2(x_{i+1}, ..., x_n), \\ & i = 1, ..., n. \\ & \mu_1, \mu_2 : \text{ group means, } \sigma^2\text{: group variance.} \end{aligned} \tag{3}$$

Table 3. Derived context of the mRNA measurements for the genes *MMP1, MMP9,* and *TIMP1*

Key	MMP1=1	MMP9=1	TIMP1=1
N87_0			×
N87_1	×		
N87_2	×		×
N87_4	×		×
N87_12	×	×	×
G205_0			
G205_1			×
G205_2		×	×
G205_4		×	
G205_12	×	×	×

Table 4. Thresholds for stimulation-dependent scaling

Stimulation	MMP1	MMP9	TIMP1
TNF	> 8.65	> 4.97	> 10.46
TGFB1	> 7.29	> 4.86	> 10.39

Thus, for each of the considered genes and for each stimulus, the ln-values $x_1, ..., x_n$ of the mRNA measurements of all patients and time points were sorted into two groups with threshold x_i.

For this trial of a scaling method, we chose the classical Ward clustering method and got meanigful results. In a further study, only minor differences and no improvements were observed when applying k-means clustering or single linkage clustering, which are also used for the clustering of gene expression data [26, Data discretisation].

In the next section we shortly introduce the main ideas concerning the conceptual representation of temporal data in transition diagrams.

2.4 Conceptual Time Systems with Actual Objects and a Time Relation (CTSOT)

In the following we represent and visualize our previously described data, namely the expression of selected genes in SFB of six arthritic patients during a period of 12 hours after stimulation. For the representation of these processes we use TCA. The reader is referred to [24] for an introduction of the notions around *Conceptual Time Systems with actual Objects and a Time relation (CTSOT).* For a more general representation of *distributed objects* the reader is referred to the paper "Applications of Temporal Conceptual Semantic Systems" by Wolff in this volume.

Now we give a short introduction to the main ideas around the notion of a CTSOT. A CTSOT is a mathematical structure describing a data table whose row entries are pairs (p,t) where p is called an *object* (interpreted for example as a patient) and t a *time granule* (interpreted for example as a time point or a

time interval). The pair (p,t) is called an *actual object*. As usual, the column entries of the data table are many-valued attributes (called *variables* in statistics). For each many-valued attribute m and each actual object (p,t), the value $m((p,t))$ is shown in the corresponding cell where the column of m and the row of (p,t) meet. A CTSOT has for each of its many-valued attributes m a conceptual scale $S_m = (G_m, M_m, I_m)$. The derived context of the CTSOT with respect to these conceptual scales is denoted by $\mathbb{K}$.

For the purpose of a clear introduction of the notion of a *state of an actual object*, the set of the many-valued attributes of a CTSOT is split into two parts, the time part and the event part, leading to a split of the data table and a split of the derived context $\mathbb{K} = (\mathbb{K}_T|\mathbb{K}_C)$ into a time part $\mathbb{K}_T$ and an event part $\mathbb{K}_C$. For each actual object (p,t) the object concept $\gamma_C(p,t)$ of the event part $\mathbb{K}_C$ is defined as the *state of the actual object* (p,t).

For the introduction of the notion of a *life track of an object* p we specify a *time relation* R_p on the set of time granules of p. In our example, all objects (here: patients) have the same time relation, namely just the "next-time-point-relation" given by the following arrows: $0 \to 1 \to 2 \to 4 \to 12$. These are represented by the arrows in the following *transition diagrams*. A transition diagram is a line diagram of the concept lattice of a part of the derived context of a CTSOT together with some arrows. An arrow leading from one object concept c to another object concept d is drawn if and only if c is the object concept of an actual object (p,t) and d is the object concept of the actual object (p,t') where $(t,t') \in R_p$, the chosen time relation on the time granules of p. The set of all object concepts $\gamma_C(p,t)$ of an object p is defined as the *life track of* p in the *state space* which is the set of all object concepts of actual objects in the event part $\mathbb{K}_C$.

Now we describe the conceptual time systems which will be evaluated in this paper.

2.5 Conceptual Time Systems for Six Arthritic Patients

We start with the full many-valued context $\mathbb{K}_0$ with 60 formal objects and 22 many-valued attributes as explained in Section 2.1. Its set G of formal objects is partitioned into the two sets G_1 consisting of the 30 TNF-stimulated cases from $N87\_0$ to $N205\_12$ and G_2 consisting of the 30 TGFB1-stimulated cases from $G87\_0$ to $G205\_12$. In this paper we shall evaluate either the 30 TNF-stimulated cases in G_1 or the 30 TGFB1-stimulated cases in G_2. That has the advantage that we can employ the usual tools for plain scaling of the many-valued subcontexts $\mathbb{K}_1$ and $\mathbb{K}_2$ induced by G_1 or G_2 to generate the two corresponding derived contexts using the threshold scaling as explained previously, namely ordinal scaling for the time, and nominal scaling for the other attributes; in addition, we can reduce the labels of the formal objects, for example from $G87\_0$ to $(87,0)$, the usual notation for an *actual object*. This labeling will be used in the following diagrams which refer either to all TNF- or to all TGFB1-stimulated cases.

All conceptual time systems in this paper will be constructed from the "TNF-subcontext" $\mathbb{K}_1$ or the "TGF-subcontext" $\mathbb{K}_2$ by selecting some of the

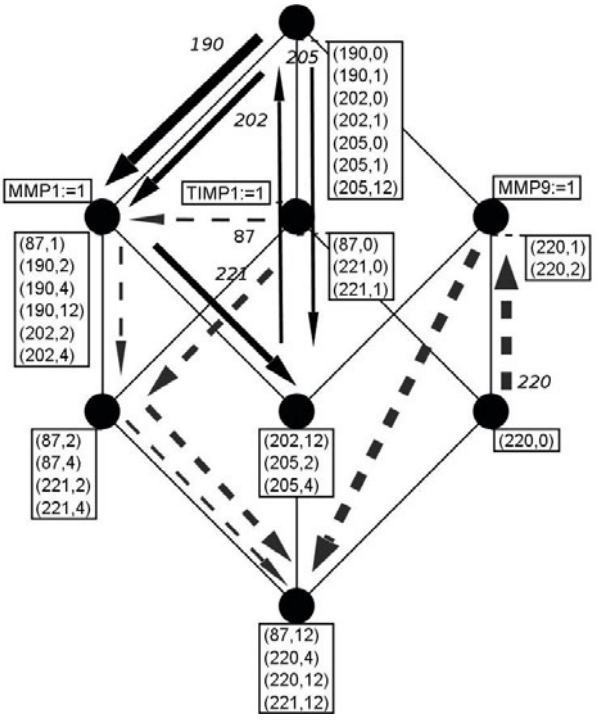


Fig. 4. Transition diagram for six patients after TNF stimulation. *Dashed arrows*: RA patients 87, 220, 221; *solid arrows*: OA patients 190, 202, 205. The first arrow of the life track of a patient is marked with the label of the patient.

many-valued attributes. The time part will always consist of the single many-valued attribute "Time". The event part will be chosen as a subset of the set of the mRNA measurements for the 18 selected genes. For each patient, the time relation is $0 \to 1 \to 2 \to 4 \to 12$.

To explain the notion of a *transition diagram* we first study the example in Figure 4. It shows the life tracks of all six patients in the concept lattice of the formal context with all $6 \times 5 = 30$ TNF-stimulated actual patients N87_0, ..., N205_12 as formal objects; its attributes are the three attributes in Table 3, i.e., MMP1, MMP9, and TIMP1, and its incidence relation is derived by the plain scaling indicated in the TNF-row of Table 4. This formal context is the derived context of the event part $\mathbb{K}_C$ of a CTSOT with the Time-column as indicated in Table 1 as its time part.

In the concept lattice of this derived context we see that all 8 possible combinations of the 3 attributes occur as intents of formal concepts; while the top concept has an empty intent, the bottom concept has all three attributes in its intent. Hence the formal object (190,0) has none of the three attributes, while (220,0) has the attributes TIMP1=1 and MMP9=1, but not the attribute MMP1=1.

We now follow the life track of patient 87. At time point 0 patient 87 has only the attribute "TIMP1=1" (see Table 3 and Figure 4). Hence the object concept of $(87, 0)$, the *state* of $(87, 0)$, is the attribute concept of "TIMP1=1". The state of $(87, 1)$ is the attribute concept of "MMP1=1". Therefore a (thin dashed) arrow is drawn from the circle of the state of $(87, 0)$ to the circle of the state of $(87, 1)$. This first arrow of the life track of patient 87 is labeled with "87". From Figure 4 one can see that patient 87 is in the same state at the time points 2 and 4, namely the concept with the intent {MMP1=1, TIMP1=1}. The life track of patient 87 ends in the bottom concept where all attributes are fulfilled, that is all three genes are expressed.

Finally we give a short technical description of the generation of the transition diagrams in this paper. At first we imported an EXCEL data table of the many-valued context of a CTSOT into the program CERNATO which is part of the DECISION SUITE by the distributor NAVICON. This data table has in its first two columns the labels for the patients (e.g., 87) and the time points (e.g., 12). Using CERNATO we scaled the many-valued attributes, and the attributes of the derived context were combined to suitable views, for example to the view represented by the line diagram in Figure 4. To include the life tracks of objects into such a line diagram, one has to save the information about the CTSOT and the views in CERNATO in an xml-file which can be imported into the program SIENA [5]. There, the life tracks of single or all objects can be easily included in the line diagram, yielding a transition diagram which can be graphically optimized by the user. In SIENA it is possible to animate the life tracks visualizing movements along the life tracks. This is of great help for the understanding of temporal effects.

Since CERNATO is not easily available any more, it is also possible to generate views with SIENA (by duplicating the context and deleting columns) and to use EXCEL or programming (e.g. in R) for scaling. If necessary, the authors will give hints.

After having discussed the conceptual representation of temporal data in transition diagrams, we now interpret some relevant diagrams from a biological and/or medical point of view.

3 Results

To evaluate our gene expression data, we focus on relevant combinations of attributes and observe the temporal behavior of the six patients with respect to the selected attributes. Clearly, the small number of patients can result only in preliminary hypotheses, which have to be compared with literature data and should be validated in future investigations. One of our main sources regarding the mutual dependency of gene expression is a literature search [26] concerning 18 genes relevant for the formation and degradation of ECM. The main results of this literature analysis are shown in Figure 5. In this graphic the solid arrows indicate an induction of the final gene by the initial gene of the arrow. A dashed

line indicates that the initial gene represses the expression of the final gene; for example, TNF represses SMAD7. In the following, we will compare the results in Figure 5 with the hypotheses deduced from the transition diagrams.

3.1 Two Destructive Proteins and Their Antagonist

The matrix-metalloproteases (MMPs) degrade the ECM of cartilage and bone. Therefore, they are important mediators of the destructive effects in rheumatic diseases. TIMP1 proteins, secreted by fibroblasts and other cells, bind to MMPs and neutralize their effects. We chose MMP1 and MMP9 due to their typical behaviour and in order to highlight the differences to TIMP1. MMP13 was omitted, since it was only expressed in RA patients at late time points. The expression of these genes was analyzed after stimulation with TNF (Figure 4) and TGFB1 (Figure 6), respectively.

Stimulation with TNF. In the case of stimulation with TNF (Figure 4), the top concept (for which all three genes are off) appears to be the initial state for all OA patients (190, 202, 205), but for none of the RA patients. It is further striking that the life tracks of all OA patients remain in the "upper area" where TIMP1 is off, whereas those of the RA patients (87, 220, 221) fall under the attribute concept of TIMP1=1, with the exception of the two intermediate states $\gamma_C(87,1)$ and $\gamma_C(220,1) = \gamma_C(220,2)$. Coarsely stated: no OA state has TIMP1 on, but most of the RA states have it. This could be a hint on an increased constitutive expression of the protective protein TIMP1 in RA. However, it is known that higher levels of TIMP1 in RA are compensated by considerably upregulated MMP expression [2], [7]. Moreover, the original experimental data (compare Table 1) reveal only small differences. They may be relevant, but should be checked by comparisons to other results. Thus, the downregulation of TIMP1 at 1 h for RA patient 87 and for 220 at 1 h and 2 h becomes more important. Finally, Figure 4 shows that MMP1 and MMP9 production increases remarkably for nearly all transitions; that is in accordance with the activation arrows TNF $\rightarrow$ MMP1 and TNF $\rightarrow$ MMP9 in Figure 5.

All these facts and observations indicate an accelerated disease progression in RA. The downregulation of TIMP1 is a surprising new observation regarding studies reporting a TNF-dependent TIMP1 upregulation [4] (compare the activation arrow TNF $\rightarrow$ TIMP1 in Figure 5). However, Alsalameh et al. [2] reported that TIMP1 was induced by TNF only in OA, but not in RA patients. Our corresponding (negative) result for RA - partly also for OA - supports the new opinion challenging the TIMP1 induction by TNF in RA.

Stimulation with TGFB1. In Figure 6, the top concept is the initial state of all OA patients, as in the case of TNF stimulation. This also represents an internal validation, since the measured value at 0 h for both stimuli (i.e., TNF and TGFB1) is derived from two independent experiments with cells from the same batch of patient cells, yielding almost identical results. For patient 190, MMP1 is upregulated after 4 h. For patient 202, MMP9 is on only after 12 h. The

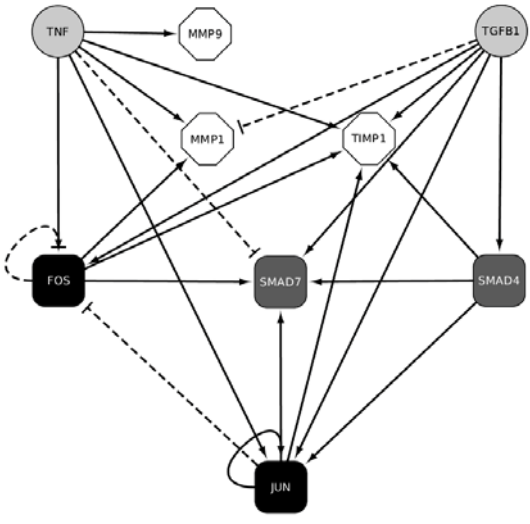


Fig. 5. Knowledge based network of the genes regulating MMP1, MMP9, TIMP1, SMAD4 and SMAD7. *TNF, TGFB1*: extracellular signaling proteins; *JUN, FOS, SMAD4*: transcription factors; *SMAD7*: inhibiting protein of *SMAD4*; *MMP1, MMP9*: matrix destructing proteins; *TIMP1*: antagonist of *MMPs*. *Solid arrows*: induction, *dashed lines*: repression of gene expression.

life track of patient 205 (thin arrows) shows an increasing behavior for MMP9: off - off - on - on - on at the 5 observation times; at the end, also MMP1 is expressed (bottom concept). In all RA states TIMP1 is on. That is complementary to the fact in Figure 4 that TIMP1 is off in all OA states. That means, TIMP1 is always expressed at a high concentration by the RA cells after TGFB1 stimulation, and at a low concentration by the OA cells after TNF stimulation. This fact underlines the bias towards a slightly enhanced expression of TIMP1 protein in RA, independently of the stimulus. Surprisingly, in Figure 6 only patient 221 shows the expected downregulation of MMP1 (and MMP9) following TGFB1 stimulation [17]. For the patients 202 and 220, MMP1 remains off, for patient 87 on, and for the patients 190 and 205 it is even upregulated. Therefore, we could not confirm inhibiting effects of TGFB1 on matrix metalloproteases, as reported elsewhere.

Finally, we observe from Figure 6 that there is no formal concept where only MMP1 and MMP9 are expressed. They are expressed together at the state represented by the bottom concept. Thus, the following implication holds:

$$\text{MMP1} = 1, \text{MMP9} = 1 \longrightarrow \text{TIMP1} = 1 \tag{4}$$

Our biological interpretation of this implication in the case of TGFB1 stimulation is: If MMP1 and MMP9 are both expressed, their effect is balanced by TIMP1.

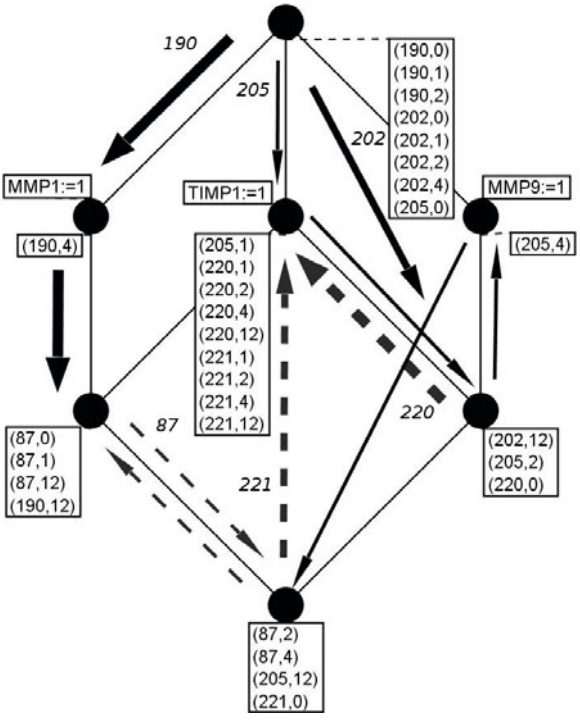


Fig. 6. Transition diagram for six patients after TGFB1 stimulation. *Dashed arrows*: RA patients (87, 220, 221); *solid arrows*: OA patients (190, 202, 205).

3.2 Transcriptional Regulation of TGFB1 Effects

The purpose of the stimulation with TGFB1 was to obtain further insight into the mechanisms regulating potential TGFB1 effects in the pathogenesis of RA. We want to understand the behaviour of the main mediators of TGFB1 effects, the transcription factors SMAD3, SMAD4 and SMAD7. SMAD3 and SMAD4 act together, and SMAD7 is able to inhibit these proteins. This is an effect of signal transduction, not directly of gene expression, i.e., phosphorylated SMAD7 protein inactivates SMAD3-SMAD4, so that they are not able to bind to the DNA and to regulate the expression of other genes. Since no knowledge is available concerning the regulation of SMAD3 gene expression, we identified SMAD3 and SMAD4 in the network in Figure 5.

In Figure 7 we focus on the development of SMAD3, SMAD4 and SMAD7 and observe a remarkable effect: SMAD7 is upregulated in all patients after 1 hour and - with the exception of patient 220 - downregulated after 4 hours. After 12 hours also SMAD3 is downregulated (with the single exception of patient 205). SMAD4 is nearly always on (with the exceptions of patient 87 for all time points, and patient 190 at 0 h). There are no clear differences between RA and OA patients.

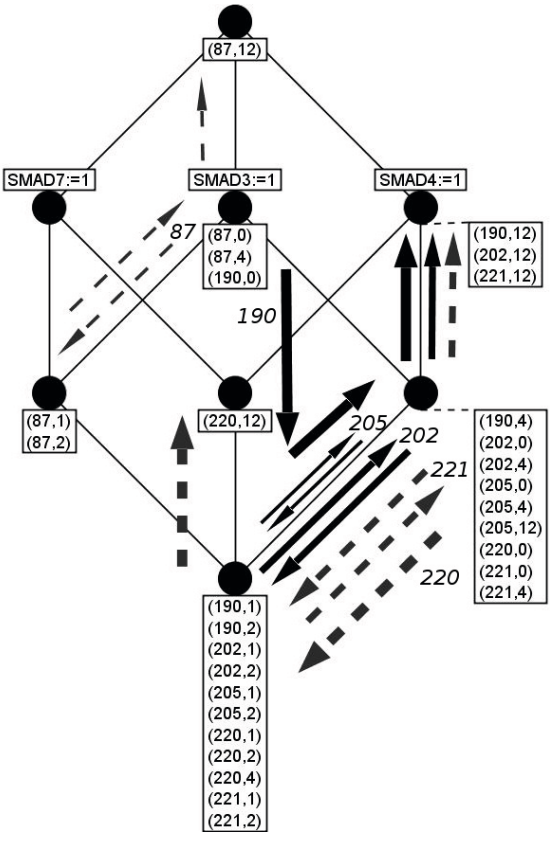


Fig. 7. TGFB1-stimulated patients and their transcription factors SMAD3, SMAD4 and SMAD7. Main effect: SMAD7 upregulation after one hour, SMAD7 downregulation after 4 hours.

From Figure 5 we see that JUN and FOS are known to induce the expression of SMAD7. Now, in order to confirm this knowledge or to find other transcription factors that might be responsible for the SMAD7-effect in Figure 7, we generate the diagram in Figure 8 with ETS1, FOS, JUN, JUNB, JUND and NFKB1 as formal attributes. Since the SMAD7-effect shows an upregulation of SMAD7 after 1 hour we search for transcription factors which are upregulated at time point 0. To visualize that, we have indicated all initial states in Figure 8 by black circles.

Now we can easily see that JUNB is on in none of the initial states; NFKB1 is on only in the initial state of patient 87; FOS is on only in the initial states of 2 patients, 205 and 220; JUN is on only in the initial states of 3 patients, 190, 202, and 205; ETS1 and JUND are on in the initial states of 4 patients, 202, 205, 220, and 221. We mention, that SMAD4 is on in the initial states of 4 patients, while SMAD3 is on in the initial states of all 6 patients (Figure 7).

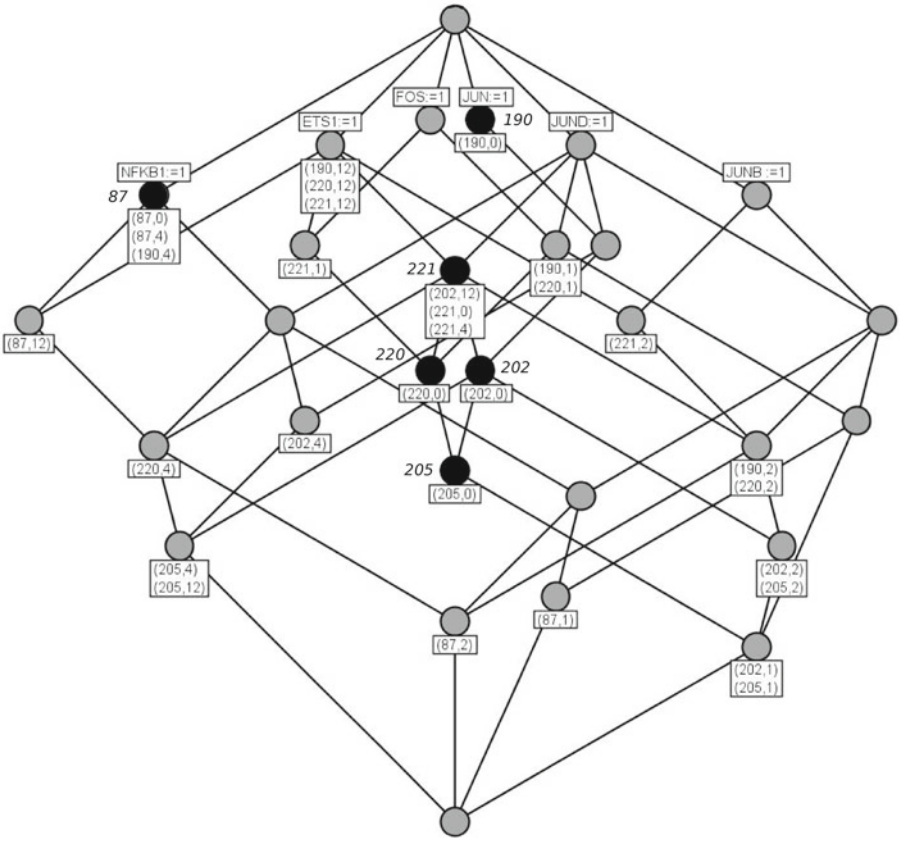


Fig. 8. Possible influences on SMAD7 upregulation at 1 h: initial states of the observation (black circles)

Thus, ETS1 and JUND could be involved in the SMAD7 upregulation 1 hour later, whereas JUN is only confirmed as an inducer in the case of the OA patients. As will be confirmed in the next paragraph, it is the most plausible hypothesis that - at least at the beginning of the time series - TGFB1 upregulates SMAD7 by SMAD3 and SMAD4.

This is in accordance to the known fact that a TGFB1 signal can activate the transcription factor SMAD4. Signaling processes within a cell are much faster than gene expression (i.e., 10 minutes versus approximatively 1.5 hours), so that a TGFB1-SMAD4 effect after 1 hour is possible.

As could be seen in Figure 7, SMAD7 was mostly downregulated after 4 h. We make a parallel investigation to the previous one and regard the transcription factors at 2 h (Figure 9).

The smallest superconcept of the 2 h states is

$$(\{(202,1), (205,1), (87,2), (190,2), (202,2), (205,2), (220,2), (221,2)\}, \{\text{ETS1, JUNB}\}),$$

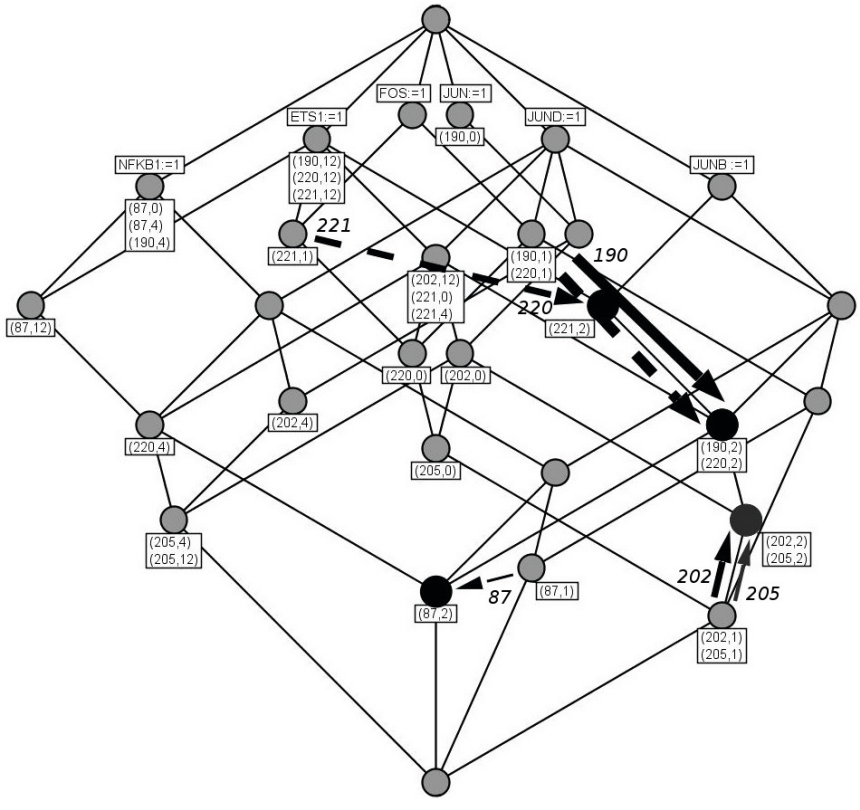


Fig. 9. Possible influences on SMAD7 downregulation after 4 hours: transitions 1 h → 2 h

i.e., at this time all patients express ETS1 and JUNB (and also two patients at 1 h). Hence, these two transcription factors could have an influence on the downregulation of SMAD7 at 4 h. However, ETS1 was on at 0 h for 4 patients and then had no repressing effect on SMAD7. Since JUNB was off at the initial state of all patients, it is a more convincing candidate as a repressor of SMAD7. This assumption would be a new finding and is worth of supplementary experimental inquiry. It is supported by data about the dependency of SMAD7 transcription on AP1 complexes containing JUN. This transcription is suppressed in the case of functional inactive JUN mutants [15]. Since JUNB is discussed as a molecule characterized (at least in part) by inhibitory effects on transcription [14], JUNB may indeed negatively influence SMAD7 transcription, if it replaces JUN in the respective AP1 complexes.

The only known inhibitor of SMAD7 in fibroblasts - NFKB1 - was expressed at 2 h in one patient only. Moreover, NFKB1 could not be activated by TNF, which is completely lacking in the TGFB1 stimulated cells (the dashed inhibitory arrow TNF → SMAD7 in Figure 5 stands for the whole signal transduction pathway

via NFKB1). Following the analysis of Figure 8, JUND could be an activator of SMAD7. Now we see that the absence of JUND was not the reason for SMAD7 downregulation, since JUND is on for 5 patients at time point 2 h. This also questions its role as an inducer of SMAD7 at 1 h, so that the positive effect of SMAD4 on SMAD7 transcription remains as the best explanation, as mentioned above.

4 Discussion

With the aid of Temporal Concept Analysis, we were able to derive new biomedical hypotheses from the data. New candidates for SMAD7 induction and repression were identified, whereas the known inducers FOS and JUN as well as the repressor NFKB1 were not confirmed to be relevant under the given conditions. The differences in TIMP1 gene expression for RA and OA patients were difficult to understand, but literature studies helped to arrive at a plausible interpretation that partly supports and partly challenges previous knowledge. Finally, our visualization of the data accentuated a contradiction to the generally supposed MMP downregulation by TGFB1. These new findings and hypotheses now await experimental confirmation.

What are the reasons for the inspiration and activation which we got during our conceptual investigation of gene expression processes? Using FCA it is possible to represent the data without any loss of information. In the case that we wish to restrict our view to an expert-oriented, specified granularity one can do that in many meaningful ways using conceptual scaling. That leads to visualizations in line diagrams of concept lattices which activate the semantics of the expert for a suitable interpretation of the original data.

The main advantage of the transition diagrams in Temporal Concept Analysis lies in the fact that they generalize the classical visualization of trajectories in physics. That is based on a generalization of the notions of space and time and the introduction of a general notion of temporal objects, states, transitions, and life tracks. Since states in a CTSOT are object concepts of actual objects, the state space is a subset of the concept lattice of the chosen view. Hence the life tracks of temporal objects can be represented in transition diagrams of suitable expert oriented views.

The previously mentioned Boolean networks do not have a notion of a temporal object. Each Boolean network together with an initial state yields a CTSOT with a single temporal object, "the system", which has as its life track just the set of all future states of the given initial state. Therefore, this approach is not suited for the visualization of our gene expression data with several patients which have been represented as temporal objects.

For the gene expression data, the transition diagrams proved to be a very efficient tool for understanding complicated temporal data. Clearly, it may be difficult to draw a good transition diagram even with the help of the computer program SIENA which is a part of the general FCA suite TOSCANAJ [5]. Interpreting the transition diagrams, the expert's free and associative thinking was

supported effectively. We got a deeper insight into the data and were activated to discuss the gene expression processes with respect to possible causes of the observed effects.

For future investigations, the actual computer programs should be improved to admit much more life tracks, to draw nested transition diagrams, and to include labels of temporal objects at the first transition arrow of a life track. Clearly, there are many possible wishes for animations of processes which should be realized in the computer programs for TCA.

In summary, our method stands for a way "back to the roots" in bioinformatics. Against the tendency to apply large scale quantitative data analyses and complicated algorithms to high throughput measurements yielding hardly interpretable results, our method activates the whole knowledge of experimental scientists concerning specific gene interactions and is adapted to their way of thinking. It is based on a mathematical theory of concepts that represents the original information with respect to the chosen granularity by a meaningful and structured visualization.

References

1. Alberts, B., et al.: Molecular Biology of the Cell, with CD-ROM, 5th rev. edn. Taylor & Francis, London (2008)
2. Alsalameh, S., et al.: Preferential induction of prodestructive matrix metalloproteinase-1 and proinflammatory interleukin 6 and prostaglandin E2 in rheumatoid arthritis synovial fibroblasts via tumor necrosis factor receptor-55. J. Rheumatol. 30(8), 1680–1690 (2003)
3. Barwise, J.: The Situation in Logic. CSLI, Lecture Notes 17, Stanford (1989)
4. Barwise, J., Perry, J.: Situations and Attitudes. MIT Press, Cambridge (1983); Deutsch: Situationen und Einstellungen: Grundlagen der Situationssemantik. De Gruyter, Berlin (1987)
5. Becker, P., Hereth Correia, J.: The ToscanaJ Suite for Implementing Conceptual Information Systems. In: Ganter, B., Stumme, G., Wille, R. (eds.) FCA 2005. LNCS (LNAI), vol. 3626, pp. 324–348. Springer, Heidelberg (2005)
6. Birkhoff, G.: Lattice Theory, 3rd edn. Amer. Math. Soc., Providence (1967)
7. Cunnane, G., Fitzgerald, O., Beeton, C., Cawston, T.E., Bresnihan, B.: Early joint erosions and serum levels of matrix metalloproteinase 1, matrix metalloproteinase 3, and tissue inhibitor of metalloproteinases 1 in rheumatoid arthritis. Arthritis. Rheum. 44(10), 2263–2274 (2001)
8. Dasu, M.R., Barrow, R.E., Spies, M., Herndon, D.N.: Matrix metalloproteinase expression in cytokine stimulated human dermal fibroblasts. Burns 29(3), 527–531 (2003)
9. Ganter, B., Wille, R.: Conceptual Scaling. In: Roberts, F.S. (ed.) Applications of Combinatorics and Graph Theory to the Biological and Social Sciences, pp. 139–167. Springer, Heidelberg (1989)
10. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Heidelberg (1999); German version: Springer, Heidelberg (1996)
11. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. J. Theor. Biol. 22(3), 437–467 (1969)

12. Motameny, S., Versmold, B., Schmutzler, R.: Formal Concept Analysis for the Identification of Combinatorial Biomarkers in Breast Cancer. In: Medina, R., Obiedkov, S.A. (eds.) ICFCA 2008. LNCS (LNAI), vol. 4933, pp. 229–240. Springer, Heidelberg (2008)
13. Smolen, J.S., Stein, G.: Therapeutic Strategies for Rheumatoid Arthritis. Nature Reviews 2, 472–488 (2003)
14. Schütte, J., et al.: Jun-B inhibits and c-Fos stimulates the transforming and transactivating activities of c-Jun. Cell 59(6), 987–997 (1989)
15. Quan, T., He, T., Voorhees, J.J., Fisher, G.J.: Ultraviolet irradiation induces Smad7 via induction of transcription factor AP-1 in human skin fibroblasts. J. Biol. Chem. 280(9), 8079–8085 (2005)
16. Ward, J.H.: Hierarchical grouping to optimize an objective function. J. Amer. Stat. Assoc. 58, 236–244 (1963)
17. White, L.A., Mitchell, T.I., Brinckerhoff, C.E.: Transforming growth factor beta inhibitory element in the rabbit matrix metalloproteinase-1 (collagenase-1) gene functions as a repressor of constitutive transcription. Biochim. Biophys. Acta. 1490(3), 259–268 (2000)
18. Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Rival, I. (ed.) Ordered sets, pp. 445–470. Reidel, Dordrecht (1982)
19. Wolff, K.E.: Temporal Concept Analysis. In: Mephu Nguifo, E., et al. (eds.) ICCS 2001. International Workshop on Concept Lattices-Based Theory, Methods and Tools for Knowledge Discovery in Databases, pp. 91–107. Stanford University, Palo Alto (2002)
20. Wolff, K.E.: Transitions in Conceptual Time Systems. International Journal of Computing Anticipatory Systems CHAOS 11, 398–412 (2002)
21. Wolff, K.E.: Interpretation of Automata in Temporal Concept Analysis. In: Priss, U., Corbett, D.R., Angelova, G. (eds.) ICCS 2002. LNCS (LNAI), vol. 2393, pp. 341–353. Springer, Heidelberg (2002)
22. Wolff, K.E.: 'Particles' and 'Waves' as Understood by Temporal Concept Analysis. In: Wolff, K.E., Pfeiffer, H.D., Delugach, H.S. (eds.) ICCS 2004. LNCS (LNAI), vol. 3127, pp. 126–141. Springer, Heidelberg (2004)
23. Wolff, K.E.: Turing Machine Representation in Temporal Concept Analysis. In: Ganter, B., Godin, R. (eds.) ICFCA 2005. LNCS (LNAI), vol. 3403, pp. 360–374. Springer, Heidelberg (2005)
24. Wolff, K.E.: States, Transitions, and Life Tracks in Temporal Concept Analysis. In: Ganter, B., Stumme, G., Wille, R. (eds.) Formal Concept Analysis - Foundations and Applications. LNCS (LNAI), vol. 3626, pp. 127–148. Springer, Heidelberg (2005)
25. Wolff, K.E.: States of Distributed Objects in Conceptual Semantic Systems. In: Dau, F., Mugnier, M.-L., Stumme, G. (eds.) ICCS 2005. LNCS (LNAI), vol. 3596, pp. 250–266. Springer, Heidelberg (2005)
26. Wollbold, J., Huber, R., Pohlers, D., Koczan, D., Guthke, R., Kinne, R., Gausmann, U.: Adapted Boolean network models for extracellular matrix formation. BMC Systems Biology 3, 77 (2009)

From Published Expression and Phenotype Data to Structured Knowledge: The Arabidopsis Gene Net Supplementary Database and Its Applications

Denis Ponomaryov[1], Nadezhda Omelianchuk[2], Victoria Mironova[2], Eugene Zalevsky[2,3], Nikolay Podkolodny[2], Eric Mjolsness[4], and Nikolay Kolchanov[2,3]

[1] Institute of Informatics Systems, Novosibirsk, Russia
[2] Institute of Cytology and Genetics, Novosibirsk, Russia
[3] Novosibirsk State University, Novosibirsk, Russia
[4] Institute for Genomics and Bioinformatics, University of California, Irvine, USA

Abstract. We report on the development progress of the AGNS (Arabidopsis gene net supplementary) database and a AGNS-based information system for automation of research on the morphogenesis of Arabidopsis thaliana (L.), a well-known model plant in system biology.

1 Introduction

At present, a large volume of information on plant genetics is available only in the form of scientific papers and hence represented as non-structured data. This appears to be an obstacle for building mathematical models of genetic processes and planning new experiments. As a rule, the information is summarized in reviews in which a lot of significant details are inevitably omitted. On the other hand, the scientific data available in publications are already quite enough for developing tools for analysis and prediction of regulation of numerous genetic processes. For solving this problem, a number of information resources have been developed (TAIR, AREX, Plant Ontology, and others), which summarize data on morphogenesis of plants. Still, there is increasing need in systems that could provide access to all possible details from published papers, as well as efficient tools for analysis and reasoning on these data. In our work, we focus on the developmental processes of Arabidopsis thaliana (L.), the well-known model plant in biology. To automate the research, we have developed the AGNS (Arabidopsis gene net supplementary) database (http://wwwmgs.bionet.nsc.ru/agns) and a specialized information system equipped with a set of tools for scientific analysis of data on gene expression and phenotypes of the plant. In the paper, we report on the progress of our work, from collecting and pre-processing data to developing tools for data analysis.

The paper is organized as follows. In Section 2 we describe the basic structure of the AGNS data and in Section 3 give an overview of terminological systems

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 101–120, 2011.

developed around the AGNS database. Section 4 contains a survey of tools for data analysis as well as some implementation details; Section 5 concludes the paper with an outlook onto further research.

2 The Structure of the AGNS Data

The AGNS database has been developed as a structured data warehouse of experimental results on gene expression and phenotypes of the wild type, mutant and transgenic plants of Arabidopsis thaliana. The database has been built on the basis of annotations of published papers describing these experiments. It is important to notice that the data structure has not been determined in advance due to the nature of the source information and the data scheme underwent several corrections. Finally, the following four modules have been logically defined in AGNS (we define their structure by predicates with the corresponding names).
Gene expression database (AGNS_ED) contains facts of the form:
$AGNS\_ED(Gene, Anatomy\_Element, Developmental\_Stage, Express\_Level, IsAbnormal)$, where $Anatomy\_Element$ is an organ, tissue, or cell and $Developmental\_Stage$ is a developmental stage of a particular $Anatomy\_Element$. An example of this would be the following fact:

$$AGNS\_ED(AG, Floral\_Meristem, FDS3, Present, False)$$

stating that expression of gene AG is normally present in floral meristem at "flower developmental stage 3".

Phenotypic abnormalities database (AGNS_PD) contains data of the form: $AGNS\_PD(Genotype, Anatomy\_Element, Developmental\_Stage, Abnormality)$
For instance, $AGNS\_PD(clv1-1, Floral\_Meristem, FDS3, Enlarged)$ means that the floral meristem at stage FDS3 is enlarged in mutant plants homozygous for the clv1-1 allele.

Publications database (AGNS_RD) contains the full list of publications on experiments on the basis of which AGNS_ED and AGNS_PD have been built. Each paper in AGNS_RD is provided with a reference to the PubMed [12] database and a description of growing conditions and ecotype of A. thaliana that was used for control in each experiment.
Sequence database (AGNS_SD) contains information on genes, their mutant alleles, and transgenic constructions. The localization of mutation in nucleotide sequences and transgene fused components are described in details, with references to source papers.

3 Terminological Systems

As far as facts in AGNS_ED and AGNS_PD contained names for anatomical elements of the plant and their developmental stages, two separate taxonomies (controlled vocabularies on plant anatomy, morphology, and developmental stages) were built around the annotated data. A detailed intensional description of concepts including qualitative and quantitative data was also provided, with references to source papers. The standard hierarchy of concepts (via the "is_a" and

"part_of" relations) was implemented in each taxonomy. Besides, a separate terminological system was built, which contained not only the names of anatomical elements and stages, but also information about the development sequence of the plant. In the following, we call this terminological system AGNS ontology.

3.1 AGNS Ontology

While taxonomies have been developed mainly for data annotation, the aim in constructing the ontology was to apply it for algorithms of analysis of expression and phenotypic data. The benefits of using formal ontologies for processing experimental data are recognized in bioinformatics [2, 4, 6]. In this section, we describe the results of formalizing plant anatomy in development by example of navigating in the AGNS database. We believe this use case to be rather simple, yet illustrative for our work. We show that by having a concrete practical task to be solved, the choice of formalization becomes well-founded and easier to evaluate, than in some abstract case. In the following, we briefly explain the task of navigation in a database via an ontology and the problem of formal representation of plant anatomy in development. Next, we introduce the formalization that was used for building the Arabidopsis ontology and present a discussion of it.

We formulate the task of navigating the database via an ontology as consisting of the following two subtasks:

- query the database in ontological terms (concepts);
- use relations between ontological concepts to broaden/narrow data extraction from the database.

A typical example of navigation in AGNS_ED would be like this: if we are interested in the expression of a gene G in some organ X of the plant, then obviously we would like to know its expression in all sub-organs of X. Or we would like to restrict ourselves to only certain parts of X or certain developmental stages of X. This implies that the ontology should contain the necessary relations between anatomical elements and developmental stages. It turns out that in order to formulate queries and manage extractions of data from the database, one needs to query the ontology itself. In our case, two typical queries, to which the ontology should provide answers are:

(Q1) X is an anatomical element, find all elements Y belonging to X;
(Q2) S is a developmental stage, find all stages earlier/later than S.

The main problem in formally describing a plant's anatomy is connected with the following general fact: if X and Y are two anatomical elements, then X belongs to Y at developmental stage S_i does not necessarily imply that X belongs to Y at another stage S_k, $i \neq k$. The number of organs in a plant may change within development and moreover, an anatomical element may have different direct containers (i.e. elements to which it directly belongs) at different stages.

Example 1. *In this example we consider the development of a leaf primordium. The leaf primordium has shoot apical meristem (SAM) as a direct container in stages earlier than P1. Later its direct container is the shoot apex (Figure 1).*

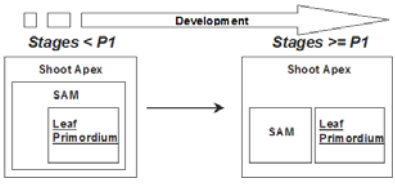


Fig. 1. Change of direct container

Example 2. *This is a more simple example; here we consider development of a floral primordium. As the primordium develops, sepal, petal and stamen organs arise in it (Figure 2).*

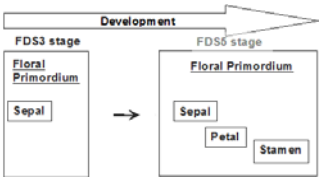


Fig. 2. Change in the number of contained organs

From these examples one can see that the relation "belongs to" is temporal. With respect to database navigation, it directly affects what expression data should be extracted for a queried plant/organ/tissue developmental stage. Suppose that this relation is represented statically. If we are interested in what genes show expression in some organ X at a stage S, we would also like to know expression in sub-organs or tissues of X at S. If X is floral primordium, then according to the "belongs to" relation, we would get expression data on all anatomical elements that belong to it. Even if we are interested in an early developmental stage, we would get expression data for sepals, petals, stamens, etc. at all stages of their development. This is an undesirable result, of course. Fortunately, Arabidopsis development is well studied, "what-where-when" is known, and stages can be considered as discrete. Eventually, the information we would like to be expressed in the ontology are statements of the kind:

"Anatomical element X at stage S_X exists in anatomical element Y at stage S_Y" (I)

We only need to represent this concept by a combination of suitable binary predicates due the implementation requirements we describe further. Before we define a formal structure for representing statements of the form (I) some preliminary remarks should be made. From a theoretical point of view we consider formal ontology as a set of sentences in the language of the first-order logic. But for practical purposes we will restrict ourselves to a decidable fragment of this language and to formulas of a special kind, which, in particular, will be reflected by the choice of the OWL language for implementation. In a formal ontology we distinguish a signature that is a set of predicate symbols in our case and axioms that restrict possible interpretations of these symbols. Throughout this section we will use the term "binary predicate" and "relation" to denote the same things.

The signature of the core of the AGNS ontology contains two unary and three binary predicates:

$$\begin{aligned} &Anatomy\_Element \\ &Developmental\_Stage \\ &Has\_Developmental\_Stage(Anatomy\_Element \times Developmental\_Stage) \\ &Before(Developmental\_Stage \times Developmental\_Stage) \\ &Occurs\_In(Developmental\_Stage \times Developmental\_Stage) \end{aligned} \tag{II}$$

For brevity in particular for writing the axioms of the ontology we will assume that binary predicates are defined on Cartesian products of those sets, that are defined by unary ones and are mentioned informally in the parenthesis.

Let us denote the statement (I) by predicate $ExistsIn(X, S_X, Y, S_Y)$, i.e. the predicate is true, whenever statement (I) holds for X, S_X, Y, S_Y. We define this predicate by the following combination of binary predicates that were introduced above:

$$\begin{aligned} ExistsIn(X,S_x,Y,S_Y) \longleftrightarrow\ & Has\_Developmental\_Stage(X, S_X) \\ & \wedge Has\_Developmental\_Stage(Y, S_Y) \wedge Occurs\_In(S_X, S_Y) \end{aligned} \tag{1}$$

The definition is illustrated in Figure 3.

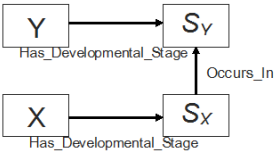


Fig. 3. Representation of the main statement

One should notice that there are no direct relations between anatomical elements. Instead, the relation $Occurs\_In$ serves for inclusion of anatomical elements into each other.

The axioms that we include in the ontology and that restrict interpretation of the predicates are:

$$\forall x \exists s(Anatomy\_Element(x) \rightarrow Has\_Developmental\_Stage(x, s)) \tag{2}$$

$$\forall s \neg(Occurs\_In(s, s) \vee Before(s, s)) \tag{3}$$

$$\begin{aligned} \forall x, s_1, s_2 \neg(&Has\_Developmental\_Stage(x, s_1) \wedge \\ &Has\_Developmental\_Stage(x, s_2) \wedge Occurs\_In(s_1, s_2)) \end{aligned} \tag{4}$$

$$\forall s_1, s_2 \neg(Before(s_1, s_2) \wedge Occurs\_In(s_1, s_2)) \tag{5}$$

$$\forall s_1, s_2, s_3(Before(s_1, s_2) \wedge Before(s_2, s_3) \rightarrow Before(s_1, s_3)) \tag{6}$$

$$\forall s_1, s_2, s_3(Occurs\_In(s_1, s_2) \wedge Occurs\_In(s_2, s_3) \rightarrow Occurs\_In(s_1, s_3)) \quad (7)$$

In our implementation the sentences 2-5 define constraints to be checked on instantiation of the ontology with concrete facts from plant development. Sentences 6 and 7 state that relations $Before$ and $Occurs\_In$ are transitive and actually define rules to infer new facts. For clarity we would like to mention also the following constraint:

$$\forall x,s(Anatomy\_Element(x) \wedge Has\_Developmental\_Stage(x, s) \rightarrow \\ \neg\exists y(Anatomy\_Element(y) \wedge Has\_Developmental\_Stage(y, s) \wedge \neg(x = y)))$$

It is true of the current facts contained in the AGNS ontology. However we did not put it in the core to allow the same stages for particular anatomy elements, which may be identified as different, but have similar developmental processes. Trichomes on cauline leaves and trichomes on rosette leaves are examples of such anatomical elements. By allowing this, of course, one should remember that according to the definition of $ExistsIn$ if for some x, y, and s

$$Has\_Developmental\_Stage(x, s) \wedge Has\_Developmental\_Stage(y, s)$$

holds and also

$$Has\_Developmental\_Stage(z, t) \wedge Occurs\_In(t, s)$$

is true for some z and t, then z (at stage t) exists both in x and y (at stage s).

For the task of navigating in the AGNS database and ease of implementation we extend the signature of the ontology by additional binary predicates. First, we define non-transitive relations $Before\_Directly$ and $Occurs\_In\_Directly$ as "sub-relations" of the initial $Before$ and $Occurs\_In$ to support navigation by levels:

$$\forall s_1, s_2(Before\_Directly(s_1, s_2) \longleftrightarrow (Before(s_1, s_2) \wedge \\ \wedge \neg\exists s(Before(s_1, s) \wedge Before(s, s_2) \wedge \neg(s = s_1) \wedge \neg(s = s_2)))) \quad (8)$$

$$\forall s, x, s_1, s_2 \neg(Has\_Developmental\_Stage(x, s_1) \wedge \\ Has\_Developmental\_Stage(x, s_2) \wedge \\ Before\_Directly(s, s_1) \wedge Before\_Directly(s, s_2)) \quad (9)$$

$$\forall s_1, s_2(Occurs\_In\_Directly(s_1, s_2) \longleftrightarrow (Occurs\_In(s_1, s_2) \wedge \\ \wedge \neg\exists s(Occurs\_In(s_1, s) \wedge Occurs\_In(s, s_2) \wedge \neg(s = s_1) \wedge \neg(s = s_2)))) \quad (10)$$

Indeed, if we used the transitive $Before$ relation for processing query (Q2), we would get all stages before/later the source stage simultaneously. A similar thing would happen if we used $Occurs\_In$ for unfolding an anatomical structure by query (Q1). Typically, we wish to restrict the level of detail, when viewing anatomical structure. A simple example would be if we want to see what tissues an organ consists of, but do not care about the cellular level.

Next we add relations $Occurs\_In\_Start$ and $Occurs\_In\_End$ with the rule

$$\forall s, s_1, s_2, s_3 (Occurs\_In\_Start(s, s_1) \wedge Occurs\_In\_End(s, s_3) \wedge Before(s_1, s_2) \wedge Before(s_2, s_3) \rightarrow Occurs\_In(s, s_2)) \quad (11)$$

to ease instantiation of our ontology. The reason for this was the prevailing manual input of facts in the ontology, as it is much easier to select an interval of developmental stages, than selecting stages one by one.

Finally we define relation $Develops\_From$ via the relations introduced above to represent development of one anatomy element into another:

$$\forall X, Y, S_X, S_Y (Has\_Developmental\_Stage(X, S_X) \wedge Has\_Developmental\_Stage(Y, S_Y) \wedge \neg(X = Y) \wedge Before\_Directly(S_X, S_Y) \rightarrow Develops\_From(Y, X)) \quad (12)$$

The core concepts defined by unary predicates in (II) together with relations defined by the mentioned binary predicates and axioms (2)-(12) make up the formal ontology, which we call *ontology of morphogenesis of plants*.

3.2 Discussion of the Ontology

Supported Queries. Let us get back to the queries (Q1) and (Q2) that were presented in Section 3.1 as the initial points of our study. Considering the formal structures introduced above the procedure of finding an answer to (Q2) should be clear, however that for (Q1) needs to be explained in detail. First, the query itself should be reformulated. We could already see that if X is an anatomy element, it makes sense to ask of what elements it consists, but it is more correct to ask of what elements X consists at a concrete stage S_X. From this perspective the procedure of unfolding anatomical structure of X at stage S_X is the following. If level-by-level unfolding is needed, then the first target set is

$$\{\ A_i \mid Anatomy\_Element(A_i) \wedge Has\_Developmental\_Stage(A_i, S_{A_i}) \wedge \wedge Occurs\_In\_Directly(S_{A_i}, S_X))\ \},$$

while the second is

$$\{\ B_j \mid Anatomy\_Element(B_j) \wedge Has\_Developmental\_Stage(B_j, S_{B_j}) \wedge \wedge Occurs\_In\_Directly(S_{B_j}, S_{A_i}))\ \}$$

and so forth to the needed level of detail. We illustrate this also by Figure 4. If the level of detail is not a matter of concern, then the target set is

$$\{\ A \mid Anatomy\_Element(A) \wedge Has\_Developmental\_Stage(A, S_A) \wedge \wedge Occurs\_In(S_A, S_X)\ \}$$

As a summary we provide the list of queries most relevant to the task of database navigation that are supported by our ontology:

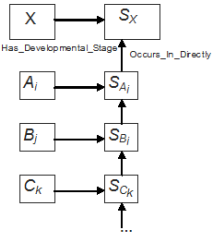


Fig. 4. Unfolding anatomical structure of X at stage S_X

- S is a developmental stage; find stages earlier/later than S;
- A is an anatomical element, S its developmental stage; find elements belonging to A at S;
- A is an anatomical element, S its developmental stage; find elements containing A at stage S;
- A is an anatomical element; find anatomical elements, from which it develops;
- A is an anatomical element; find anatomical elements developing from A;
- A and B are anatomical elements; find stages that A undergoes while being a part of B;
- A and B are anatomical elements; find stages that A undergoes while having B as its part;
- A and B are anatomical elements; is A a part of B at any developmental stage of B (or A)? (i.e. does A happen to be a part of B at all?)

Instantiation of the Ontology. Other facts that help to evaluate the AGNS ontology originate from the experience of its instantiation with real data from arabidopsis development studies. Textual information about arabidopsis anatomy and development was extracted from published papers and entered into the ontology in the form defined by formal constructs from the previous section. During this process we have encountered two main problems regarding incompleteness of information, namely:

1) stages were not defined for all anatomy elements we would like to include in the ontology;
2) development of lots of anatomy elements was described in the manner "X has the following properties, when Y is at stage S" (e.g. "LDS2 leaf shows meristematic divisions throughout the mesophyll" [5], but these descriptions were given only for some stages of Y.

This has led to the need of creating 'artificial' developmental stages for anatomy elements. Let us illustrate this by the following example.

Example 3. *Suppose we want to describe the development of tissue in organ X of a plant. Let us assume that five stages of organ development are known, but only two developmental stages for the tissue are distinguished: T_1 for the first stage of X and T_2 for the next three stages of X. We can describe this fact by the auxiliary Occurs_In_Start and Occurs_In_End relations in the ontology (as shown in Figure 5).*

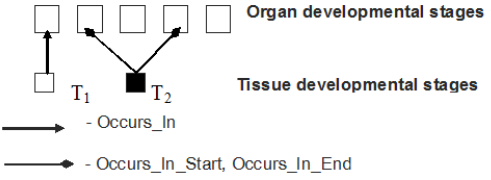


Fig. 5. Development of tissue in an organ

Now suppose that we also wish to represent cell development in this process and that cell development is described relative to organ X developmental stages (Figure 6).

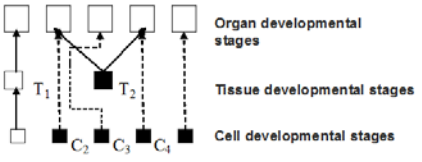


Fig. 6. Development of cells in an organ

The first decision on how to represent this in our formal model will probably be to define cell developmental stages as occurring directly in the corresponding organ developmental stages (via the *Occurs_In_Directly* relation). This will not contradict the general understanding of plant structure in the sense that cells are indeed part of organs. But this will lead to an incorrect unfolding of the anatomical structure of X at its developmental stages starting from the second one. Even if we define stages C_2 , C_3 and C_4 as occurring directly in T_2 (Figure 6), this will lead to loss of information, as we have assumed that cell developmental stages are described relative to organ developmental stages in this example. That is why we need to introduce so called "artificial" stages for the tissue. In particular, we split T_2 into three stages and add stage T_5 that was not present before (Fig. 7).

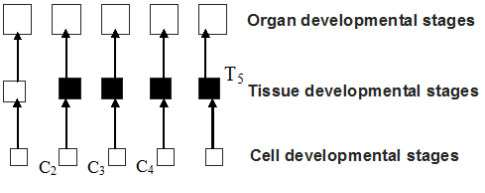


Fig. 7. "Artificial" stages for tissue are created

Even though these stages may not be distinguished in reality, we need to have them in the ontology to provide the correct unfolding of organ anatomy structure.

3.3 Implementation of the Ontology

In the implementation we were guided by two objectives: to have a widely supported expressive ontology language with full reasoning capabilities and to use an ontology editor with rich import-export and visualization functions. This resulted in the choice of the OWL language and the Protégé ontology editor (http://protege.stanford.edu). The problem of representing rule-like axioms was not considered as a significant one, because the number of such axioms in our ontology is small. Thus, it is possible to rewrite them manually into the language of the needed inference engine. In our case we used the Algernon engine for Protégé. Sentences 2-5,8,9,10 were implemented as constraints, sentence 11 as a forward chaining rule, and 12 – as backward chaining rule in the Algernon language. Relations $Before\_Directly$ and $Occurs\_In\_Directly$ defined in 8,10 were represented by the subPropertyOf construct of OWL and transitivity properties 6,7 – as the TransitiveProperty construct.

Another point to mention here is how we have considered the problem of loops in development of anatomy elements. Loops occur in many situations, like for example, continuous leaf formation in plant growth. As the ontology has been constructed mostly by manual input, it would have made no sense to enter manually the facts that each developmental stage of a leaf occurs in every developmental stage of vegetative growth. To describe such situations we define a special class $Universal\_Stage$ in our ontology with the rule, which we present here informally: "for any stages s and u, if u is an instance of $Universal\_Stage$ and u is an instance of some class C and s occurs in u, then s occurs in every instance of class C". For example, take C as the class of the vegetative developmental stages of a shoot apex and assume that universal stage u belongs to C and s is a leaf developmental stage. Then putting the statement that s occurs in u in our ontology will generate facts that s occurs in every instance of C. One can see that this rule is defined by the second-order logic formula

$$\forall s, t, u, C(Universal\_Stage(u) \wedge C(t) \wedge C(u) \wedge Occurs\_In(s, u) \rightarrow Occurs\_In(s, t)),$$

which we can not implement, of course, in a first-order reasoner, when C is not fixed. That is why we used the Protégé scripting environment and made a small program that determined the list of classes C (in our notation) having universal stage as an instance, and executed this rule for each class from the list.

In general, the problem of representing plant anatomy in development was reduced to ordering and inclusion of developmental stages. The AGNS ontology has a small core, yet with strong expressive capabilities. It became possible to define the $Develops\_From$ relation for anatomical elements not as a separate one, but via the core concepts of the ontology. Information about Arabidopsis development taken from publicly available articles was presented in a formal way. Each developmental stage in the ontology was provided with textual description with a reference to the source article. The implemented ontology is a deductive database consisting of facts about plant development and inference rules. The proposed formalization was developed in order to apply it for algorithms of analysis of expression and phenotypic data, which are given in Section 4.

3.4 Comparison with Existing Formalizations

The Plant Ontology (PO) [9–11] information resource called simultaneously controlled vocabulary and ontology has been developed for anatomy and development of flowering plants. The arabidopsis ontology provided by PO is the internationally recognized terminological system created by the collaborative effort of specialists in biology and bioinformatics. Thanks to coordination of knowledge on flowering plants, it became possible in particular, to unify key terms regarding to anatomical elements and their developmental stages. We used PO to verify that the names for anatomy elements and developmental stages in our ontology do not contradict with the internationally accepted terms. At the same time, we could not apply PO for the task of database navigation and data analysis directly. As we could see above, for this task the temporal aspect of the $ExistsIn$ relation between anatomy elements needs to be taken into account.

Disregarding temporality of some key relations that appear in biomedical ontologies has been already mentioned in papers [1, 13, 14]. In our work, we avoided using modalities, but this in turn has made us introduce a special signature of ontology; it differs significantly from that of PO. In this sense, our work is closer to the approach of Aitken, described in [1], where a detailed analysis of the current research on formal representation for biological concepts, like species, sex, and developmental stage is given. The author introduces an axiom for describing the part-whole relation for a part-class and a whole-class at a concrete developmental stage. This can be viewed as a formalism for representing statements like (I) in Section 3.1, but only when stage S_X is not taken into account.

The paper by Smith et al. [14] can be recommended as a reference one for knowledge engineers working in bioinformatics. In this paper, the authors introduce axiomatic semantics for most of the relations that are central for biomedical ontologies. A temporal version is given not only for the key relations such as "instance of", "part of", but also proposed for mereotopological ones [15]. However, it is not clear from the paper, if there exists a temporal axiomatization for mereotopology. The considered relations can be used for describing plant anatomy in development. However, when implemented in a straightforward way, they may cause difficulties. In the case of the $ExistsIn$ relation, if we fixed a set of time granules (stages) of development of an organ, then we would have to list all anatomy elements that are part of this organ for each time granule. This is of course, an undesirable point in modeling. The more correct decision (for representing plant development) is to describe all developmental stages that occur at each of time granules, provided that each stage is sufficiently annotated. We believe this modeling decision to be more natural, judging from how plant development is actually described by experimenters.

3.5 Functions of Terminological Systems

With the development of the terminological systems, the AGNS database has become the core of the information system allowing for annotating papers on experiments on gene expression and phenotypic abnormalities of the Arabidopsis plant, as well as for analyzing these experimental data.

- The terminological systems are connected with data fields of the AGNS_ED and AGNS_PD databases and allow for unambiguous choice of names for anatomical elements and developmental stages when entering new data.
- The navigation in AGNS_ED and AGNS_PD is implemented via the terminological systems.
- Each concept in the terminological systems is provided with a detailed textual description with a reference to the source article. The terminological systems contain names for anatomical elements and developmental stages that comply with the standard names approved by the Plant Ontology Consortium (PO) [9]; thus they can be used for integration of AGNS with other databases on plant genetics [8]. In comparison with PO, the terminological systems of AGNS are more detailed and not only contain intensional description of every concept, but also include fine-grained concepts not present in PO, which allow for a more detailed annotation of papers and a higher 'resolution' for studying plant developmental processes. For instance, the terminological systems of AGNS describe 58 anatomical elements in the arabidopsis embryo compared to only 15 in PO.

The terminological systems of AGNS contain knowledge about the elements of morphogenesis, their types and relationships between them – all in the form needed for automated processing of the underlying experimental data. They are used in conjunction with the AGNS_ED and AGNS_PD modules for describing and integrating experimental data on gene expression and phenotypic abnormalities and are necessary for logical comparison of results of the both types of experiments.

4 Input and Processing of Data in AGNS

The input of data obtained from annotation of scientific papers is made via the input system implemented as a Java application. This system supports the following features:

- annotation of papers and saving the extracted data in the AGNS database;
- interaction with the terminological systems: adding new concepts, editing existing ones, and using them for input of data into the AGNS_ED and AGNS_PD databases;
- syntactic analysis of the input data with respect to terminological systems.

The AGNS database is constantly updated with new data, which are obtained from annotation of scientific papers. After a new paper is processed, the experimental data from this paper soon become available in AGNS. At first, data are put into an auxiliary database, then they are verified by an expert biologist and transferred into the online database. In the current implementation, AGNS is based on Berkley DB XML. The extensibility of the system is provided by using reflexive and modeling technologies of Java in XML. This allows for maximal flexibility of the basic components of the system, as well as their applicability in a wide range of tasks. The access to AGNS is made via Web-interface,

which is implemented as a Java-applet. The interface visualizes terminological systems of AGNS and provides a choice of queries to the database. The queries are implemented in XQuery language and are optimized for execution in Web-environments. The query execution results are represented in the form of tables. The AGNS_ED module supports the following queries:

- What is the expression of a gene X in the wild type of the plant? The output is the summarized data from all the annotated papers describing normal expression of gene X at developmental stages and domains of the plant.
- Which genes show expression in an anatomical element X at a stage S? If X or S is not specified, then the output is a list of genes with the corresponding names for anatomical elements/develomental stages, at which expression is present.
- What genes show coexpression with a gene X? The output is a list of genes having the same expression patterns with X and the list of anatomical elements/developmental stages at which this takes place.
- Mutations or transgenes of which genes change expression of a gene X? What are these changes and at what stages/in which anatomical elements do they occur?
- Expression of which genes is changed in mutant/transgenic plants for a gene X? What are these changes, at what stages/in which anatomical elements do they occur? If the name for a mutation/transgene for X is not specified, then all mutant and transgenic plants of the gene are considered. The user may also specify an anatomical element and/or developmental stage as a filter.

The following queries are supported by the AGNS_PD module:

- In which anatomical elements and at what stages do mutations/transgenes of a gene X cause phenotypic abnormalities, and what are these abnormalities?
- Mutations/transgenes of which genes cause the same abnormalities as for a gene X? At what stages/in which anatomical elements do they occur?
- Mutations or transgenes of which genes lead to phenotypic abnormalities of an anatomical element X? What are these abnormalities and at what developmental stages do they occur?

4.1 Tools for Data Analysis

As soon as AGNS has accumulated a significant amount of experimental data on plant genetics, it becomes possible to solve a number of important biological tasks with the help of data analysis tools. Comparing data on normal development of the plant, phenotypes of mutants, time and place of gene expression allows for finding the developmental stage and anatomical element, in which a phenotypic abnormality is predetermined. Moreover, it becomes possible to define the role of particular genes in the development of the abnormality. For this purpose, the AGNK (Arabidopsis GeneNet Knowledge Base) was developed, which comprises algorithms for finding predetermination stages of abnormalities and discovering relationships between abnormalities.

Finding Predetermination Stages
The first algorithm is a data processing scenario for AGNS implemented in the XSLT language. It uses information from the both modules AGNS_ED and AGNS_PD for finding stages of predetermination of a given abnormality. The scenario is executed in three steps:

Step 1. Defining regulatory processes in gene expression. In the AGNS taxonomy all developmental stages are subdivided into several basic classes such as: embryo development, seedling, axillary SAM development, leaf development, flower development, and ovule development. At first, all stages preceding organ development are selected from the taxonomy of developmental stages. For these stages, all records in AGNS_ED are extracted, which have the value of Express_Level field equal to "switch on", "switch off", "increased", or "decreased". These records correspond to regulatory processes connected with gene expression. All these records are temporarily saved and those with the value of Express_Level not equal to "switch on" are marked with 1. The result is a collection of tuples of the form:

$$\langle Gene, Anatomy\_Element, Developmental\_Stage, Express\_Level, Marking \rangle.$$

Step 2. Filtering phenotypic data. For each gene selected at the first step AGNS_PD is queried for the list of all phenotypic abnormalities in plants carrying mutations or transgenes of this gene. The tuples obtained at the first step for transgenic plants of the queried gene get marked with 2. The resulting data are temporarily saved as a collection of tuples of the form:

$$\langle Gene, Allele, Anatomy\_Element, Developmental\_Stage, Abnormality, Marking \rangle.$$

Step 3. Merging data. The tables obtained at the steps 1 and 2 are processed according to special rules defined by biologists. Based on the complex hierarchy of anatomical elements, the pairs of organs (a list of comparable concepts) have been defined, with respect to which the tuples in the both tables are compared. If a tuple has the value of *Anatomy_Element* equal to any of the organs in a pair, then the tuple is saved in AGNK as the output of the processing scenario.

The AGNK data are visualized by a Java-applet in the Web-interface accessible at http://wwwmgs.bionet.nsc.ru/agns/agnkapplet . Along with the output information of the processing scenario, the interface provides access to terminological systems of AGNS and to a database containing schematic pictures of abnormalities (see Figure 8). Thus, AGNK allows for analysis and visual representation of gene regulation of morphogenesis of the Arabidopsis plant.

Discovering Relationships between Abnormalities
The second algorithm developed for AGNK aims at discovering potential relationships between phenotypic abnormalities of Arabidopsis. It takes a description of phenotypic abnormality in the form of the AGNS_PD tuple as input and generates a graph of relations of this abnormality to other abnormalities of mutant and transgenic phenotypes described in AGNS_PD. Having analyzed papers on

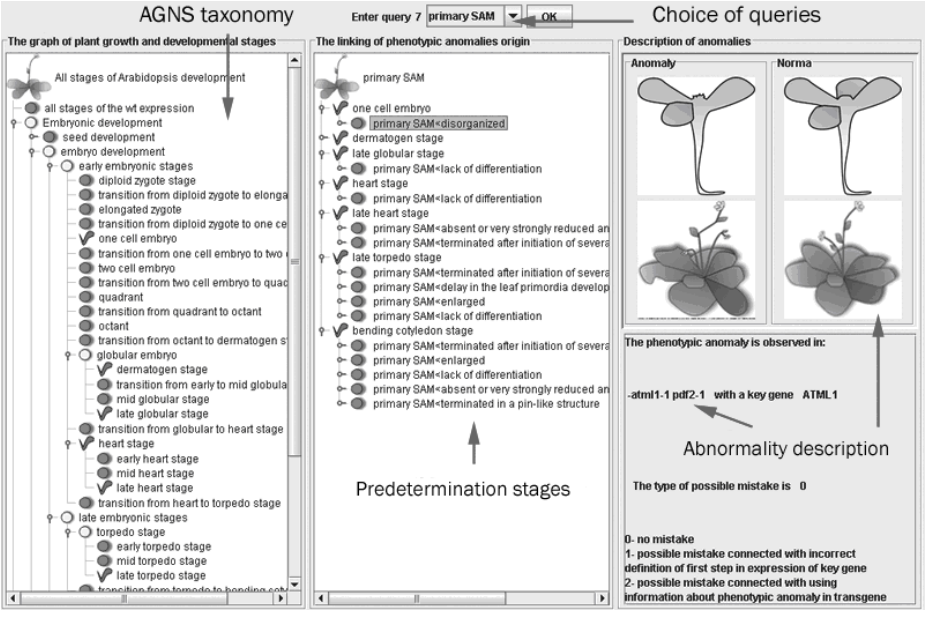


Fig. 8. The AGNK interface. In the upper part: choice of queries by the name of organ. The leftmost column: stages of normal plant development; in the middle: stages of predetermination of abnormalities (marked with a tick); the column to the right: a detailed description of the abnormality.

phenotypic effects of mutations in Arabidopsis, we came to the conclusion that experimental results described in these papers can be represented by sets of probabilistic implications of the two sorts:

- In allele/transgene A at growing conditions C abnormality Ab_1 of strength S was present in X percent of the observed cases;
- Y percent of plants with abnormality Ab_1 also had abnormality Ab_2.

The first statement is an implication between the conditions and the caused abnormality, while the second one is an implication between abnormalities implicitly stating that they are related to each other. The facts of the first sort are always mentioned in papers, however that of the second sort are rather rare. One of our aims was to establish connections between the results of different experiments published in different papers and to reconstruct relations between abnormalities on the basis of the integrated data of the AGNS database.

The algorithm operates on the data of the AGNS_PD module, AGNS ontology, and AGNS abnormality classifier. The classifier is a hierarchy of all types of abnormalities that occur in the AGNS_PD tuples. The names for abnormalities are classified according to general characteristics such as “abnormal color”, “abnormal form”, “not developed”, etc. For some of them, a more detailed

classification is given, e.g., "abnormal size: increased size", "abnormal size: decreased size". We have determined six relations of interest between abnormalities; below we give their informal, intuitive definition. Note that here we understand abnormality as a pleotropic effect observed for a concrete anatomical element at its concrete developmental stage.

"Same as": two abnormalities are the same, if they represent the same phenotypic changes, but are named differently (e.g., in different research groups). By phenotypic changes we mean here changes in morphology and development of the plant.
"Alternative to": we call two abnormalities of the same anatomical element alternative to each other, if they represent possible ways of abnormal organ development occurring in different percentages of cases. For example, in zll mutants shoot apical meristem may be absent or very strongly reduced and not restored at germination or developed into pin-like structure or terminated after initiation of several leaves [7].
"Consequence of": an abnormality is a consequence of another one, if it is the result of development of this abnormality within time. For instance, the enlarged shoot apical meristem causes an increased number of lateral organs.
"Blocked by": an abnormality blocks another abnormality of another anatomical element, if the second one is not present, when the first abnormality is observed.
"Constituent to": we call abnormality constituent to another one, if its phenotypic change makes up the phenotypic change of the second abnormality. For example, the increase of trichomes branching may be constituent to an increase in leaf pubescence.
"Inverse to": two abnormalities are inverse to each other, if they present opposite phenotypic changes, e.g., narrow or wide leaves.

In the following, we will give formal definitions for these relations, which will represent informal ones in a narrower, yet precise sense. For example, following the informal definition of "alternative to", one can conclude that those abnormalities that are formally assumed to be the same, can well represent alternative abnormalities (independent events). However, in our formal definitions the *same_as* and *alternative_to* abnormalities are disjoint sets. The formal definitions will be inference rules for the above-mentioned relations. We suggest to consider the inferred relations just as a view of the AGNS_PD data at a higher level of abstraction we are going to use next for analyzing phenotypes. A statement that one abnormality is a consequence of another abnormality is a very strong one that needs a thorough justification and analysis, if not proved experimentally. That is why it is important to understand that the algorithm can only suggest potential relationships between abnormalities, which should then be evaluated by an expert.

To explain the algorithm, we introduce a formal definition of abnormality, which will correspond to the basic AGNS_PD data structure. Consider the four sets A, E, S, and T, where

- A is a set of identifiers for genotypes of the Arabidopsis plant;
- E is a partially ordered set of anatomical elements (cells, tissues, organs and structural elements such as specially distinguished layers or zones; the whole plant is also considered as anatomical element), with the order $\rhd$ defined as follows: for any $e_1, e_2 \in E$ we have $e_1 \rhd e_2$ iff e_2 develops from e_1;
- S is a partially ordered set of developmental stages of anatomical elements with the order $>$ defined as follows: for any $s_1, s_2 \in S$ we have $s_2 > s_1$ iff s_1 is before s_2 (in time);
- T is a set of types of phenotypic abnormalities.

Define the relation $\succ \subseteq E \times S \times E \times S$ with the property that for any $e_1, e_2 \in E$ and $s_1, s_2 \in S$ we have $\langle e_1, s_1, e_2, s_2 \rangle \in \succ$ iff e_1 at developmental stage s_1 exists in e_2 at stage s_2. Following the level of abstraction, at which abnormalities are presented in AGNS_PD, we define phenotypic abnormality as a 4-tuple: $N = \langle G, e, s, t \rangle$ with $G \subseteq A$, $e \in E$, $s \in S$, and $t \in T$.

AGNS_PD consists precisely of the collection of such 4-tuples labeled by textual names (short descriptions of abnormalities extracted from papers). It follows immediately from our definition of abnormality that two different names denote the same abnormality in AGNS_PD, if they correspond to the same 4-tuples. Note that this is not the only rule the algorithm uses to identify the same abnormalities.

One can compare two abnormalities $N_1 = \langle G_1, e_1, s_1, t_1 \rangle$ and $N_2 = \langle G_2, e_2, s_2, t_2 \rangle$ by considering all possible set-theoretical relations between the ground sets G_1, G_2 and also the relations between the elements e_i, s_i, t_i, $i = 1, 2$. The possible relations between ground sets we consider are: $G_1 = G_2$, $G_1 \cap G_2 = \varnothing$, $G_1 \subset G_2$, $G_2 \subset G_1$, and finally $G_1 \cap G_2 \neq \varnothing$ with $G_1 \cup G_2 \supset G_i$, $i = 1, 2$. The corresponding anatomical elements and stages can be equal, incomparable, or connected by the relations $\rhd, >, \succ$ defined above. As abnormality types are organized in the hierarchy, they can be equal, incomparable, or can belong to the same parent.

We have defined 17 possible cases of difference between two abnormalities $N_1 = \langle G_1, e_1, s_1, t_1 \rangle$ and $N_2 = \langle G_2, e_2, s_2, t_2 \rangle$ that make sense in our context. For each case we have formulated rules for inferring relations between abnormalities. On one hand, these rules are formal definitions we ascribe to the relations between abnormalities given informally above. On the other hand, they define the algorithm for identifying relationships between abnormalities. While formulating these rules, we took into consideration several factors connected with the underlying data of AGNS_PD, namely:

- possible incompleteness of data (not every mutant/transgenic plant is presently studied and not all experimental data available are present in the AGNS database);
- errors by annotating papers (different abnormalities could be taken as being the same by error and saved as one record in AGNS_PD – we call this wrong factorization);

- probabilistic nature of the original data, as explained in the beginning of this section;
- existence of redundant regulatory pathways in the plant (e.g., an abnormality can be recovered at later developmental stages).

We are not able to give all the 17 rules in this paper, that is why we formulate below only some of them for each of the relations between abnormalities. Each rule is defined for a pair of tuples $N_1 = \langle G_1, e_1, s_1, t_1\rangle$ and $N_2 = \langle G_2, e_2, s_2, t_2\rangle$. For convenience of reading, the condition of each rule is written in two (or more) lines: in the first one a condition on the relations between the sets $G_i, i = 1, 2$ and elements $e_i, s_i, i = 1, 2$ is specified; further lines contain additional conditions and conclusion of the rule.

Rules for inferring *SameAs* **and** *AlternativeTo* **relations**

C1.1. $G_1 \cap G_2 \neq \varnothing$, $e_1 = e_2$, $s_1 = s_2$.
R.1.1.1. IF $t_1 = t_2$ THEN $SameAs(N_1, N_2)$ ELSE $AlternativeTo(N_1, N_2)$.

C1.2. $G_1 \cap G_2 = \varnothing$, $e_1 = e_2$, $s_1 = s_2$.
R.1.2.1. IF for N_1, N_2 there EXISTS abnormality N_3 such that $SameAs(N_1, N_3)$ AND $SameAs(N_2, N_3)$ THEN $SameAs(N_1, N_2)$.

In the cases below we split rules with disjunctive premises into several rules for clarity. It is also necessary in one case to mention that one abnormality is *SameAs* another one and that the common part of their ground sets has a non-empty intersection with some set S. We denote this further for two arbitrary abnormalities N_1, N_2 and a set S as $SameAs(N_1, N_2)/S$.

Rules for inferring *ConstituentTo* **relation**

C2.1. $G_1 = G_2$, $\langle e_1, s_1, e_2, s_2\rangle \in \succ$.
R2.1.1. IF $t_1 = t_2 =$ "not developed", THEN $ConstituentTo(N_1, N_2)$.
R2.1.2. IF there DO NOT EXIST abnormalities N_3, N_4 such that $AlternativeTo(N_1, N_3)$ OR $AlternativeTo(N_2, N_4)$ THEN $ConstituentTo(N_1, N_2)$.

C2.2. $G_1 \subset G_2$, $\langle e_1, s_1, e_2, s_2\rangle \in \succ$.
R2.2.1. IF $t_1 = t_2 =$ "not developed", THEN $ConstituentTo(N_1, N_2)$.
R2.1.2. IF (there is NO abnormality N_3) such that $SameAs(N_2, N_3)/(G_2 \setminus G_1))$ AND (there DOES NOT EXIST an alternative abnormality for N_1) THEN $ConstituentTo(N_1, N_2)$.

+ two more rules not mentioned here.

Rules for *ConsequenceOf* **and** *BlockedBy* **relations**

C3.1. $G_1 = G_2$, $e_1 = e_2$, $s_2 > s_1$.
R.3.1.1. IF $t_1 = t_2$ THEN $ConsequenceOf(N_2, N_1)$.
R.3.1.2. IF $(t_1 \neq t_2)$ AND (there are NO alternative abnormalities for N_1, which do not have consequents) AND (there is NO abnormality $N_3 = \langle G_3, e_3, s_3, t_3\rangle$ such that $ConsequenceOf(N_2, N_3)$ AND $t_2 = t_3$) THEN $ConsequenceOf(N_2, N_1)$

C3.2. $G_1 = G_2$, $e_1 \rhd e_2$.
R3.2.1. IF (t_1 ="not developed") AND ($t_1 \neq t_2$) THEN $BlockedBy(N_2, N_1)$ ELSE IF there are NO alternative abnormalities for N_1 THEN $ConsequenceOf($ $N_2, N_1)$. + 8 more rules not mentioned here.

Rule for $InverseTo$ relation

C4.1. $e_1 = e_2$, $s_1 = s_2$.
R.4.1.1. IF ($t_1 \neq t_2$, but t_1 and t_2 have a common parent in the abnormality type hierarchy) AND (there EXIST $g_1 \in G_1$ and $g_2 \in G_2$ such that g_1 is a loss-of-function gene mutation and g_2 is a gain-of-function mutation (or vice-versa)) THEN $InverseTo(N_1, N_2)$.

The rules above (+ 10 more not present here) define the algorithm for discovering relationships between phenotypic abnormalities basing on the AGNS_PD data. On may call the relations above as complex view at these data. Indeed, if we take the relations between abnormalities as Cartesian products of the corresponding ground sets of tuples, these sets can be considered as answers to complex queries to AGNS_PD database.

5 Conclusion

The AGNS database contains detailed annotations of published experiments on arabidopsis gene activity and regulation at mRNA, protein, cell, tissue and organ levels in development. The information system developed around AGNS allows to systematically integrate and classify this heterogeneous and scattered information. The system includes taxonomies of anatomical elements and their development stages, as well as the ontology of anatomical structure in development for Arabidopsis thaliana. Each terminological system provides a different granularity at viewing experimental data in the AGNS database. The coarse-grained taxonomies are used in annotation of papers and entering information into the database. The fine-grained ontology is used for navigation in the AGNS database, as well as in tools for data analysis. We have given an overview of algorithms for finding predetermination stages of phenotypic abnormalities of the plant and discovering relationships between abnormalities. When evaluated on large data sets of AGNS, these algorithms can assist in identifying key points of abnormal development, finding parallel regulatory pathways leading to abnormalities, and discovering specific functional modules in the plant. This will allow for a better understanding of mechanisms of developmental processes and their interactions and will be a step to reconstructing the underlying gene networks. The data structure of AGNS together with the tools of data analysis allow for retrieving complex data sets necessary for the investigation of various processes of morphogenesis, from gene expression to functions of genes. We believe that the approach, methodology, and tools developed for the model plant Arabidopsis thaliana can be applied for studying phenotypes and gene expression in other organisms.

Acknowledgements

The authors thank Elliot M. Meyerowitz (California Institute of Technology, Pasadena, USA) for his valuable pieces of advice and comments on this paper. The work was supported in part by the DFG-Project COMO (GZ: 436 RUS 113/829/0-1), Russian Foundation for Basic Research (grants No. 08-04-01214-a), Siberian Branch of the Russian Academy of Sciences (integration project No. 119), Programs of RAS 22(8),23(29), and the US National Science Foundation (FIBR EF-0330786 Development Modelling and Bioinformatics).

References

1. Aitken, S.: Formalizing concepts of species, sex and development stage in anatomical ontologies. Bioinformatics 21, 2773–2779 (2005)
2. Bard, J.B., Rhee, S.Y.: Ontologies in biology: design, applications and future challenges. Nat. Rev. Genet. 5, 213–222 (2004)
3. Berardini, T.Z., et al.: Functional annotation of the Arabidopsis genome using controlled vocabularies. Plant Physiol. V. 135, 745–755 (2004)
4. Bodenreider, O., et al.: Biomedical ontologies. In: Altman, R.B., Dunker, A.K., Hunter, L., Jung, T.A., Klein, T.E. (eds.) Proceedings of Pacific Symposium on Biocomputing, pp. 76–78. World Scientific, Singapore (2005)
5. Carland, F.M., McHale, N.A.: LOP1: a gene involved in auxin transport and vascular patterning in Arabidopsis. Development 122(6), 1811–1819 (1996)
6. Karp, P.D.: An ontology for biological function based on molecular interactions. Bioinformatics 16, 269–285 (2000)
7. Lynn, K., et al.: The PINHEAD/ZWILLE gene acts pleiotropically in Arabidopsis development and has overlapping functions with the ARGONAUTE1 gene. Development 126, 469–481 (1999)
8. Mironova, V.V., Poplavsky, A.S., Ponomaryov, D.K., Omelianchuk, N.A.: Ontology of Arabidopsis Genenet Supplementary Database(AGNS): Cross references to TAIR ontology. In: Proc. Bioinformatics of Genome Regulation and Structure, pp. 209–212 (2005)
9. Plant Ontology Consortium, `http://www.plantontology.org`
10. Vincent, P., et al.: The Plant Ontology Consortium and plant ontologies. Comparative and Functional Genomics 3(2), 137–142 (2002)
11. Jaiswal, P., et al.: Plant ontology (PO): a controlled vocabulary of plant structures and growth stages. Comparative and Functional Genomics 6(7-8), 388–397 (2006)
12. The PubMed database, `http://www.ncbi.nlm.nih.gov/pubmed/`
13. Smith, B., et al.: On the application of formal principles to life science data: a case study in the gene ontology. In: Rahm, E. (ed.) Database Integration in the Life Sciences, pp. 79–94. Springer, Heidelberg (2004)
14. Smith, B., et al.: Relations in biomedical ontologies. Genome Biology 6 (2005), `http://genomebiology.com/2005/6/5/R46`
15. Smith, B.: Mereotopology: A theory of parts and boundaries. Data and Knowledge Engineering 20, 287–303 (1996)
16. TAIR The Arabidopsis Information Resource, `http://www.arabidopsis.org`

How Can Ontologies Contribute to Software Development?

Alexander S. Kleshchev

Institute for Automation & Control Processes
The Far-Eastern Branch of the Russian Academy of Sciences,
Radio str. 5, 690041 Vladivostok, Russia
kleschev@iacp.dvo.ru
http://www.iacp.dvo.ru/is/

Abstract. The aim of this article is to discuss some directions in software development where using ontologies can lead to a considerable progress. The basic concepts related to ontologies are considered. It is shown that ontologies can be a basis for domain analysis and simulation in software development. Ontologies can be used for interactive design of objects such as dances, musical compositions, user interfaces, and so on. Also using ontologies can lead to considerable progress in forming knowledge bases, maintaining expert systems and developing optimizing compilers controlled by knowledge bases.

Keywords: ontology, knowledge base, software development, analytics, interactive designing, forming knowledge bases, maintaining expert systems, optimizing compiler controlled by a knowledge base.

1 Introduction

At the present time one of the intensively studied concepts in software development is ontology [1]. There are a significant number of directions in software development where ontologies are playing a prominent role. Creation of Web-portals on the basis of ontologies and using ontologies for navigation in information arrays [2] and for intellectualizing software agents [3] are among them. However, the potential of ontologies is considerably wider. The aim of this article is to discuss some other directions in software development where using ontologies can lead to considerable progress.

2 Basic Concepts

In this article information is considered as a collection of ideas, i.e., as a meaningful concept.

A mapping of a finite set of terms $t_1, \ldots, t_m$ into the (infinite) set of all possible values V will be called a verbal representation. In other words, a verbal representation is a table consisting of two columns and m lines. The terms $t_1, \ldots, t_m$

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 121–135, 2011.

are placed in the first column and their values from the set V are placed in the second one. By this means a verbal representation is a formal language with the elementary syntax.

Information is verbalizable if it can be transmitted adequately by a verbal representation. This definition establishes a link between these meaningful and formal concepts. The word "adequately" means that on transmission of information by a verbal representation the recipient will obtain the same information from the verbal representation that the sender coded.

Various questionnaires are examples of verbal representations used in day-to-day life. An interpretation function for a signature of the predicate calculus language is an example of a verbal representation used in mathematics. A representation for a state of a computer process in terms of the program identifiers and their values is an example of a verbal representation used in programming. The last example shows that only verbalizable information can be processed by present-day computers, and this processing can be performed only in a verbal representation.

To connect the semantics with a verbal representation it is necessary to define the semantics for values (elements of the set V) and for finite sets of terms $t_1, \ldots, t_m$. This semantics can be done as follows.

The set of all possible values V is considered as the union of magnitudes. A magnitude is a set of values (a subset of the set V) closed under a finite set of operations, functions and predicates determined for this magnitude. Algebraic systems (and multi-sorted ones among them) are analogues for magnitudes in mathematics. Data types are analogues of magnitudes in programming.

The semantics of a finite set of terms $t_1, \ldots, t_m$ can be unambiguously defined by conceptualization, i.e., by the set of all verbal representations that are meaningful and have the same set of terms $t_1, \ldots, t_m$. In other words, a conceptualization is the subset of the set of all possible tables, all elements of which possess this meaningful property (have meaning). Only those who know the meaning of the terms $t_1, \ldots, t_m$ can decide whether or not a table possesses this property, i.e., the semantics of a finite set of terms is implicitly defined by all possible sensible applications of terms from this set.

An ontology (a terminological knowledge base) is the most precise specification of a conceptualization, i.e., the ontology explicitly represents the meaningful property "the table has meaning". If a conceptualization is taken as an extensional specification for a set of all sensible verbal representations with a fixed set of terms, then an ontology is an intensional specification for this set of verbal representations. It is an explicitly specified predicate which is true for all elements of the conceptualization and false for all verbal representations not belonging to the conceptualization. The ontology determines the semantics of a finite set of terms explicitly but approximately (the predicate cannot always have an explicit and exact representation). To be understandable for specialists, an ontology is usually built in the form of a finite set of definitions for terms and of ontological agreements. A definition of a term determines the denotation of the notion designated by the term (the set of all possible values for this term)

and ontological agreements determine connections among the values of different terms in verbal representations. An ontology cannot be correct or incorrect. It only specifies the meaning of a set of terms explicitly (and the meaning of terms, as is well known, is not discussed). However, the ontology itself must be defined in particular terms. They can be as follows: a chosen set of abstract terms (class, example, part, whole, attribute, value, and so on), for which their meaning is considered as known; a set of terms related to magnitudes (designations for constants, operations, functions and predicates); or a set of mathematical terms (which are unambiguously defined in mathematics by systems of axioms and mathematical definitions).

Knowledge expressed in terms $t_1, \ldots, t_m$ always relates to a proper (infinite) subset of the conceptualization, its elements possessing a meaningful property. Knowledge about this subset is the most precise specification of this subset in terms of the conceptualization (ontology) [4].

The ideas one way or the other related to a professional activity will be called professional ones. In the set of all the professional ideas some subsets can be recognized; they are called domains. Every professional idea belongs to one or several domains. A complex domain can include ideas from simpler domains.

One of the most important characteristics of a domain is its terminology (the set of terms used in the domain). A term is a word or phrase with a special meaning in a domain. Any domain can be uniquely characterized by its terminology. To understand professional ideas related to a domain is impossible without understanding the meaning of its terminology.

Based on the foregoing, a domain terminology allow us to define the domain conceptualization and ontology. The reality of a domain is the infinite set of verbal representations for information about real world fragments in terms of the domain that took place in the past, are taking place at present and will take place in the future. The elements of this set will be called situations. By this means the reality of a domain is the proper subset of its conceptualization (it is possible to imagine fragments of the world that never took place in the past, do not take place at present and will not take place in the future), all its elements possessing the meaningful property indicated above. Knowledge of a domain is knowledge about its reality [5].

3 Analytics

Development of every program (especially of an information system) begins with analyzing information about the domain which is necessary to design the program. Usually three groups of specialists deal with this activity called analytics. They are: domain experts who possess the conceptualization and knowledge of the domain and can describe applied tasks; designers of the information system who need a comprehensible domain model and task specifications in terms of the model; and analysts who are to obtain information from experts and meet designers' requirements.

At present there are two approaches to analytics. One of them uses the conceptual framework of applied mathematics and the other uses the conceptual framework forming the basis of implementation languages.

Using the conceptual framework of applied mathematics analysts (applied mathematicians) recognize objects from situations (the input data, output data and intermediate values in applied tasks), designate them by variables they introduce and associate mathematical objects (numbers, sets, functions predicates, graphs, and so on) with them. Domain notions are associated with the same variables. For each variable a range of its possible values is defined. In this case domain knowledge is most often simulated by a system of mathematical relationships among these variables, and applied tasks are reduced to mathematical tasks formulated in terms of these systems. There exist well known methods for solving many mathematical tasks.

The object-oriented analysis [6] and knowledge engineering [7] are examples of analysis methods within a conceptual framework forming the basis of implementation languages. In the first case a domain model is built as a set of classes, objects and methods; in the second case, it is built as a set of rules. In both cases the results of this analysis contain a description of methods for solving tasks of the information system under development.

Ontologies can be used as the basis for the third approach to analytics called here the ontological analysis. It uses the conceptual framework of the domain being analyzed. A domain model is built as a domain ontology model and a domain knowledge base, specifications of applied tasks being formulated in terms of the ontology model. Development of methods for solving these tasks is a step of this analysis. The goals of the ontological analysis for a domain are to find its conceptualization, to build its ontology, and to form a knowledge base in terms of this conceptualization.

The first step of the ontological analysis for a domain is searching for its conceptualization. Experts working together with analysts should make a list of terms to represent the reality as complete as necessary, and a representative list of situation descriptions given in these terms. Next, analysts together with experts try to represent verbally the situations from the list, and experts detect whether these verbal representations adequately transmit information about situations. A list of values already used is put together. During this work the lists of terms, values and situations can be supplemented. Analysts set the meanings of terms and values used and principles of adequate representation of situations. Part of this step is analysis of the value list. Every value should be assigned to a magnitude. A list of all the magnitudes used is put together. If all the situations from the list have been adequately represented as elements of the conceptualization, the magnitudes have been detected, and the meanings of all the terms and the principles of adequate representation of situations are clear to analysts, then it is said that this step is successfully completed and the domain conceptualization is recognized.

The second step of the ontological analysis is building an ontology for the conceptualization recognized. To do this, analysts together with experts should

build definitions for all terms of the conceptualization using terms related to magnitudes and those from the conceptualization already defined in the ontology. Error detection in the definitions can be performed based on the list of verbal representations for situations obtained at the previous step. If the value of a term in the verbal representation of a situation is beyond the denotation of the appropriate notion defined in the ontology, then this definition is incorrect. Finally, of special difficulty is the formulation of ontological agreements. Some of them can be proposed by experts, but there is no hope that all the ontological agreements can be obtained in this way. A more systematic way to obtain them is to put together (with experts) a list of meaningless situations that are represented verbally but do not belong to the conceptualization. An attempt to discuss the reasons why these situations do not belong to the conceptualization can lead to formulating additional ontological agreements. If it turns out that such a situation contradicts the reality ontology, then either some ontological agreements must be improved or additional ontological agreements must be formulated.

The third step of the ontological analysis for the domain is specifying knowledge to determine the domain reality as precisely as possible. Knowledge is specified in the same form as ontological agreements (in terms of the ontology). The final part of this step is detecting errors in domain knowledge. Every situation in the verbal representation should be consistent with the specified domain knowledge. Then experts make a list of situations that belong to the conceptualization but do not belong to reality. If there are such situations in the list that contradict the specified domain knowledge, then the knowledge specification should be improved. Formalization of the ontology and knowledge completes the building of the domain model.

Comparison of these three approaches to analytics allows the following conclusions to be made. The approach using the conceptual framework of applied mathematics is of limited utility since in many domains for which information systems are developed there is no tradition to use applied mathematics. In this regard, the object-oriented analysis, knowledge engineering and ontological analysis are more general-purpose methods. The approach using the conceptual framework forming the basis of implementation languages makes analysts perform part of designers' work and impose analysts' design decisions on them; designers cannot change these decisions without additional information about the domain. Clearly, the more complex a domain is, the more difficult is its analysis within this framework. Within the framework of the ontological analysis analysts are involved in building a domain model only. Their work is cut down in comparison with the previous case and so they can analyze more complex domains. In this case designers perform the whole designing and their work grows. However, in doing so they are not bound by other design decisions. Thus the areas of application are preferably complex domains (usually related to scientific activity) in which there are no traditions to apply mathematical methods. If ontology and knowledge models are represented in mathematical terms, then it is possible to perceive a certain similarity between the ontological analysis and the applied

mathematics analysis (the results of them both are represented in mathematical terms). The difference lies in the fact that in the case of the applied mathematics analysis an attempt is made to simplify a domain notion system in such a way as to build its mathematical model directly; in the case of the ontological analysis a mathematical model of ontology is built on the basis of the domain notion system (in the general case, without any simplifications) and then a mathematical model of knowledge is formed [8].

To formalize the results of the ontological analysis an ontology representation language is used. Developers of every such language have to answer the following basic questions: in which terms are ontologies defined? (as noted above, these terms should be clear to designers; they can be either terms of software engineering when designers are specialists in this field or mathematical terms when designers are assumed to have a mathematical background); what properties of ontologies can be represented? (to answer this question it is necessary to know what properties ontologies may have [9]); to what degree the complexity of ontology representation relates to the complexity of its content (for languages with a fixed syntax there may be cases when simple ideas require cumbersome formal representations). According to their purposes, ontology representation languages can be divided into languages of tools (ontologies represented in these languages are to be directly used in software designs) [10], and languages for investigating ontologies (ontologies represented in such languages are intended for publication). The languages of the first group have some implementation restrictions while the languages of the second group are free from them. The language of applied logic is an example of languages for investigating ontologies [11]. This language is extendable both syntactically and semantically. An ontology represented in it is defined in mathematical terms.

Since analysis (including the ontological analysis) of complex domains is a laborious and intellectually difficult kind of activity, it is desirable that its results could be reused. An ontology is reusable if it can be used to develop different information systems. An important class of reusable ontologies consists of real ontologies that define meanings of terms used by specialists in their professional activity. Publication of such ontologies makes results of the ontological analysis for appropriate domains accessible to designers of information systems. An ontology of medical diagnostics for acute diseases [12], ontologies of physical [13] and of organic [14] chemistry, an ontology of the X-ray fluorescence analysis [15], an ontology of computer program transformations [16] and an ontology of graphic user interface [17] are examples of real ontologies.

Metaontology is an ontology whose terms can be used to describe other ontologies. Generalizing this definition, we can speak of ontologies on various levels. Of special interest are applied (not universal) metaontologies. Chemistry is an example of a domain with a four-level applied metaontology. A wide domain is such a domain for which ontologies of all its divisions can be described in terms of the same applied metaontology of the domain, medical diagnostics being an example of a wide domain. A class of domains is a collection of domains for which their ontologies can be described in terms of the same applied ontology

of this class. Detecting applied metaontologies allows us to reveal an inner similarity among externally different domains and their divisions. Other definitions for similarity of ontologies and relations among them are considered in [8].

The ontological analysis considered above is a process of building an ontology using a finite subset of its conceptualization, i.e., a top-down process. Applied metaontologies can be used as the basis of another process, the metaontological analysis, i.e., of a process of building an ontology for a domain or its division bottom-up using an applied metaontology (of a wide domain or a class of domains). In such a manner ontologies for some divisions of chemistry and medicine have been built.

4 Interactive Designing

Let us consider a problem of building target objects of a class, each of them possessing a content and having a form. Suppose that there are two groups of specialists. They are coders (specialists in the form) and experts (specialists in the object content). Coders do not know the content every target object should have but they know how to build target objects of the required form. Experts know the content a target object should possess but they do not know how to build target objects of the required form (or building such objects is sufficiently laborious for them). During the life cycle the content of target objects can change. It may also imply changes in their form. The traditional method of solving this problem is as follows: experts explain to coders what content a target object to be built should possess. After that coders build the target object. Every time when the content of an already built target object changes, interaction between experts and coders is resumed. It is clear that the more complex the content of a target object is the more laborious and unreliable is this method. Development of computer programs (as target objects) can be an example. Here two groups of mediators are necessary, namely analysts and designers. Their aim is to provide more or less reliable and effective transmission of information between experts and coders.

Now consider a special case of this problem. Let us suppose that for every target object of a class an information structure can be defined. It is called its design (model) and meets the following requirements:

- designs of target objects are verbalizable information;
- terminology of their verbal representation is known to experts;
- an interactive program can be implemented that allows experts to build designs of target objects in this verbal representation;
- a program can be implemented to transform verbal representations of designs into target objects.

We will call this case a problem of interactive designing. It consists in automation of building and modifying target objects of a given class by experts (without coders). Designing can only be interactive if experts are not required to possess skills in using any artificial languages and they need intellectual support for

this process (so that the interactive mode considerably reduces laboriousness of designing). Assuming this, to automatize the activity of experts it is necessary to solve the following tasks:

- to work out an ontology and a knowledge base of designs, and an ontology of the design activity that meets the second requirement from those listed above;
- on the basis of these ontologies and the knowledge base, to develop an interactive CAD-system for experts to form and modify the designs of target objects;
- to develop a translator (compiler or interpreter) from the verbal representation of designs into target objects.

Although the problem of interactive designing has been successfully solved in engineering, its solution in domains far from engineering is less common and is based, as a rule, on intuition. So a few examples of interactive designing problems are given below that can be solved (or have been successfully solved) on the basis of ontologies.

Interactive editing of verbal representation controlled by ontologies is one of the problems of interactive designing. The problem consists in developing such interpreters of ontologies (metaontologies) that support experts in creating and editing information resources of various generality levels (data bases, knowledge bases, ontologies, and so on). At present such interpreters for personal computers as well as those accessible via the Internet have been implemented [18].

The problem of interactive design of dances using computers is rather poorly supported [19]. If we deal with modern solo dances, now a mirror and a video camera are basic instruments of choreographers. In the mirror they see the dance they design and perform, and using the video camera they record this performance and transmit it to their students. Animated images for dances of a certain style (computer cartoon films) are target objects in this problem, and choreographers are experts. For the implemented system of interactive design of the FUNC style dances [20] an ontology of designs consists of two parts. They are: a dance ontology with notions such as FUNC dance style, movement, pose, kinetic bar, and so on, and a human biomechanics ontology with terms of the anatomy important for choreography. The knowledge base contains restrictions of human biomechanics. The interactive CAD-system supports the graphic interface with choreographers on the basis of these ontologies and of the ontology of dance designing, of movements, and of poses of the whole body and its parts. In this process extendable libraries of previously designed dances, movements and poses can be also used. The result of designing is a dance design which can be transformed into a computer cartoon film and visualized (the whole design or its fragments) during designing as well as after its completion.

Similarly, the problem of interactive designing and analyzing musical compositions can be solved. In this case musical compositions presented in musical notation are target objects (in the form of MIDI-files that can be played by computer) and composers are experts. An ontology of musical composition designs should contain definitions of such terms as musical forms, the plan of the

composition and all of its parts, tonal plan, theme, variations, combination of voices, terms of harmony, and so on. The first step towards creation of such an ontology is made in [21]. The knowledge base should contain the description of possible forms and their plans, restrictions on tonal plans, semantic structures of themes, types of variations, possible harmonic interrelations, and so on. The task of the CAD-system is to build designs of musical compositions, transform them into the musical notation and play them by computer. The system of analysis should allow musicologists to perform analysis of musical compositions in terms of the same ontology, i.e., disassemble them into such parts that they can be assembled anew from these parts by inverse operations. Analysis of actual musical compositions can be the basic way of forming the knowledge base for such a CAD-system.

In designing user interfaces these latter are target objects. They are represented in the form of a computer program running together with an appropriate applied program. There are several groups of experts in this case. They are: experts of the domain; designers who design outward appearance of the interface; specialists in ergonomics who design a dialog scenario; software engineers who design the connection of the interface with the applied program; specialists in interface usability who evaluate the usability of the resulting design. Each of the listed groups has specific terminology defined by an appropriate ontology, and specific knowledge represented in the knowledge base of the developed CAD-system. The user interface is the part of an applied program (information system) to be modified most often because of changes in user requirements and operation conditions, and also because of modification of the applied program itself [22].

Maintenance of an information system is its adaptation to changing conditions of its operation. The more complex the information system, the more laborious its maintenance. For a developing information system a crisis can come when all the efforts of its developers are spent on its maintenance, and there are no resources to develop it further (to expand its functionality). Information systems for managing universities are examples of developing systems. For such systems changes of operation conditions consist in the fact that current business-processes can often be changed, new business-processes arise perpetually, and some old ones are not to be performed any more. Moreover, document formats and data models are changed often (especially in reforming the education system). At the same time, the functionality of such systems is very complex. It includes management of staff, training process, research, documents circulation, regular and once-only undertakings, and so on, including inspections of dormitories [23]. The problem of maintenance for such information systems can be solved if each individual system is developed as a particular system of interactive designing and a design interpreter. The system of interactive designing supports creation, development and maintenance of the information system design. This activity is performed by experts whose terminology is defined by the ontology of the system of interactive designing. The design interpreter supports operation of the information system.

5 Verbal Representation of Knowledge Bases

A knowledge base, like any information, can be represented verbally. This requires a domain terminology that is specially intended for knowledge representation such that verbal representation of the domain knowledge base using this terminology represents this knowledge adequately. An ontology defining the meaning of this terminology will be called a knowledge ontology, as distinct from an ontology defining the meaning of terms for situation representation that will be called a reality ontology. Ontological agreements of the knowledge ontology will be called knowledge integrity restrictions, and those of the reality ontology will be called situation integrity restrictions. In this case the domain ontology consists of two ontologies. They are the reality ontology and the knowledge ontology, connected by a special system of ontological agreements about the relation between knowledge and reality [5]. The ontology of medical diagnostics is an example of such an ontology [12].

The traditional way of forming knowledge bases is to get them from experts. Now there are tools for editing knowledge bases intended for experts and controlled by knowledge ontologies. We dealt with them in the previous section. Another way is inductive formation of knowledge bases using empirical data. As D. Michie noted [24], an inductively formed knowledge base can be useful only if it is understandable for specialists of the appropriate domain. Under these conditions not only these specialists can use this knowledge themselves, but they will also trust the expert system using this knowledge base. They can check conclusions of this system as well. In addition, any element of an inductively formed knowledge base should provide an explanation understandable for any domain specialist (how and wherefrom the empirical data has been obtained). In the domains where knowledge is represented verbally it is clear for domain specialists since the meaning of terms from the appropriate ontology is known to them. In [25] methods are suggested for inductive formation of knowledge bases in such domains. These methods are to solve the following task: using a training set of verbally represented situations (of examples which are possible situations and of counterexamples which are impossible situations), it is necessary to find a verbally represented knowledge base which is (the most) correct (every example is consistent with the knowledge base) and exact (every counterexample is not consistent with the knowledge base) in relation to the training set. The basic idea of these methods is using ontological agreements about the relation between knowledge and reality in the domain ontology while solving this task.

Every system based on knowledge consists of two parts, namely, a shell and a knowledge base. An expert system is a system based on knowledge where the knowledge base has a high level of competence. The traditional approach for implementing such systems (based on knowledge representation) is as follows: knowledge engineers choose or programmers implement a universal shell that is an interpreter of a knowledge representation (for example, production system); after that knowledge engineers and experts form (and then maintain) a knowledge base in this knowledge representation. If the domain knowledge base

is verbally represented, then another approach (based on a knowledge ontology model) to expert system implementation is possible: knowledge engineers together with experts form a domain ontology including the knowledge ontology; programmers develop a specialized shell that is an interpreter of the knowledge ontology; experts form (and then maintain) a knowledge base with the knowledge editor controlled by the knowledge ontology. Comparison of these two approaches shows that a demonstration prototype for a knowledge based system can be developed considerably faster using universal shells within the framework of the first approach, but development of an expert system with a knowledge base to be maintained for a long time is possible only within the framework of the second one.

If tasks which a knowledge based system should solve can be specified in terms of a metaontology (of a wide domain or of a class of domains) and knowledge bases (of domains or their divisions) have verbal representation, then this metaontology can be used as the basis for a specialized shell. Its tuning to a domain of the class or a division of the wide domain makes it a shell for this domain or division. Such a metashell should also contain a multilevel editor tuned by ontologies. At present such a metashell has been implemented for medicine; it can be tuned to its various divisions.

Usually for every knowledge based system its own ontology is worked out, its terms being used to form knowledge bases of this system. As a result, there are a lot of expert systems and knowledge bases for medical diagnostics, for example, which are not compatible with one another. However, it was the reuse of knowledge bases that was one of impetuses for distinguishing ontology as a solo concept and for its further investigation. Reuse can be achieved, above all, for knowledge bases formed in terms of real ontologies. These knowledge bases prove to be independent of expert systems for which they are intended. In this case, a specialized shell also becomes independent of these knowledge bases since any knowledge base formed in these terms can be processed by this shell.

Formation and especially maintenance of knowledge bases by a single expert or a group of experts is hardly possible in practice in the case of real expert systems since this activity is time-consuming. This results in a certain crisis of expert systems as a direction in artificial intelligence [26]. A possible way out is the collective development and maintenance of knowledge bases independent of specific expert systems. In general, this approach can be defined as collective development and maintenance of information resources independent of programs for their processing. One of the tools for supporting collective development of information resources is the Multipurpose Knowledge Bank [27]. It is a Web-system providing means of collective development of information resources of various generality levels with the IDEA-editor controlled by metainformation [18], and means of access to these resources for processing programs via the shell of the Bank.

Specialized intellectual application program packages can also be developed as systems based on verbal representation of knowledge. Universal intellectual application program packages PRIZ [28] and SPORA [29], based on frames for

knowledge representation, have not found widespread practical use. Specialized application program packages are based on appropriate domain ontologies (for example, on the ontology of physical chemistry [13]). They are intended for deducing methods to solve tasks using specifications of these tasks in terms of this ontology and knowledge base, for synthesizing programs using these methods, and for calculating the solution of these tasks using these programs and input data.

Optimizing compilers controlled by knowledge bases are another application of intellectual systems based on verbal representation of knowledge [30]. Many existing optimizing compilers have a built-in set of applied transformations and a strategy of their applications. This does not permit us to use such a compiler for experiments on studying properties of certain transformations and strategies for their applications. There have been no tools for prototyping such compilers. An optimizing compiler controlled by knowledge bases has been implemented and used for computer experiments on program optimization and transformation, to prototype optimizing compilers, and to develop means of active training of students in program optimization and transformation [30]. As the theoretical basis of such a compiler the ontology for transformation of different classes and the ontology of flow analysis already available can be used. An optimizing compiler controlled by a knowledge base consists of a syntactically controlled parser (or editor) of programs, of an interpreter of the verbal representation for the projection of a source language into the universal representation of programs, of an interpreter of the verbal representation for the application strategy for transformations, of an interpreter of the verbal representation for flow analysis methods, of an interpreter of the verbal representation for transformations, of an interpreter of the verbal representation for the projection of the universal representation of programs into a target language, and also of appropriate editors controlled by knowledge ontologies. The syntactically controlled parser (or editor) is to tune such a compiler to different source programming languages. The interpreter of the verbal representation for the projection of a source language into the universal representation of programs is to transform the program derivation tree into its universal representation. The interpreter of the verbal representation for the application strategy of transformations is to tune the compiler to a given strategy of application for transformation. The strategy determines the order of application for flow analysis methods and transformations from the knowledge base. The interpreter of the verbal representation for flow analysis methods is to make the flow analysis determined by the strategy, i.e., for enriching the program in the universal representation with calculated values of attributes. The interpreter of the verbal representation for transformations is to perform transformations determined by the strategy in an enriched universal representation. The interpreter of the projection of the universal representation of programs into a target language is to tune the compiler to different target languages. Finally, knowledge base editors are to form knowledge bases about different languages, their projections into the universal representation, strategies

of applications, flow analysis methods, sets of transformations, and projections of the universal representation into different target languages.

6 Conclusions

The preceding allows us to conclude that ontologies can be another means in our control over some kinds of complexity in software development. A new approach to analyzing and formalizing information about complex domains is proposed that is necessary for designing complex information systems. Ontologies have led to progress in developing a number of systems for interactive designing of non-technical objects. Using ontologies with verbalizable knowledge to work out problem-oriented methods of inductive formation of knowledge in verbal representation allows us to implement D. Michie's idea that inductively formed knowledge should be equally accessible to both domain specialists and expert systems not only in content but in form as well. Knowledge based systems developed as specialized shells with reusable ontologies as their theoretical base involve mechanisms to transmit maintenance of these systems from programmers to domain experts and specialists. Ontologies have also allowed us to create a transformation machine with a changeable set of transformations. Systematic use of ontologies for software development is naturally complemented by means of supporting collective development of knowledge bases and other information resources.

Acknowledgements

The research was supported by the RFBR, the grant "Control of conceptual metaontologies, ontologies, knowledge and data in intelligent software", and the Far Eastern Branch of the Russian Academy of Sciences, the grant "Development of control systems of knowledge bases with multiple access".

References

1. Uschold, M.: Knowledge Level Modeling: Concepts and Terminology. The Knowledge Engineering Review 13(1), 5–29 (1998)
2. What is ontology? Frequently asked questions, `http://www.alphaworks.ibm.com/contentnr/semanticsfaqs`
3. Wayner, P.: Free Agents. Byte 3, 105–114 (1995)
4. Kleshchev, A.S., Shalfeeva, E.A.: Classification of Ontology Properties. Ontologies and their Classifications. Scientific and Technical Information Series 2. 9, 16–22 (2005) (in Russian)
5. Kleshchev, A.S., Artemjeva, I.L.: Mathematical Models of Domain Ontologies. Int. J. Information Theories & Applications 14(1), 35–43 (2007)
6. Booch, G.: Object-Oriented Analysis and Design. Addison-Wesley Publishing Company, Reading (1994)
7. Waterman, D.A.: A Guide to Expert Systems. Addison-Wesley Publishing Company, Reading (1986)

8. Kleshchev, A.S., Artemjeva, I.L.: A Mathematical Apparatus for Domain Ontology Simulation. Logical Relationship Systems. Int. J. Information Theories & Applications 12(4), 343–351 (2005)
9. Shalfeeva, E.A.: Classification of Ontology Properties. Ontology Properties and their Classification. Scientific and Technical Information SeriesM2. 11, 9–16 (2005) (in Russian)
10. Corcho, O., Gómez-Pérez, A.: A Roadmap to Ontology Specification Languages, http://www.cs.man.ac.uk/~ocorcho/documents/ekaw00_CorchoGomezPerez.pdf
11. Kleshchev, A.S., Artemjeva, I.L.: A Mathematical Apparatus for Domain Ontology Simulation. An Extendable Language of Applied Logic. Int. J. Information Theories & Applications 12(2), 149–157 (2005)
12. Kleshchev, A.S., Moskalenko, P.M., Chernyakhovskaya, M.Y.: An Ontology Model for Medical Diagnostics. Scientific and Technical Information SeriesM2. P.1. 12, 1–7 (2005); P.2. 2, 19–30 (2006) (in Russian)
13. Artemjeva, I.L., Tsvetnikov, V.A.: An Ontology Fragment of Physical Chemistry and its Model. Electronic Journal "Investigated in Russia" 3, 454-474 (2002) (in Russian), http://zhurnal.ape.relarn.ru/articles/2002/042.pdf
14. Artemjeva, I.L., Vysotsky, V.I., Reshtanenko, N.V.: A Domain Ontology Model (by the Example of Organic Chemistry). Scientific and Technical Information SeriesM2. 8, 19–27 (2005)
15. Artemjeva, I.L., Miroshnichenko, N.L.: An Ontology model for the X-Ray Fluorescence Analysis. Informatics and Control Systems 2, 78–88 (2005) (in Russian)
16. Knyazeva, M.A., Kupnevich, O.A.: An Ontology Model for Optimizing Sequential Programs. Scientific and Technical Information SeriesM2. P. 1. 2, 17–21 (2005); p. 2. 4, 14–22 (in Russian)
17. Gribova, V.V., Tarasov, A.V.: An Ontology Model for Graphic User Interface. Informatics and Control Systems 1, 80–90 (2005) (in Russian)
18. Kleshchev, A.S., Orlov, V.A.: Computer Knowledge Banks. A Universal Direction in Solving the Problem of Editing Information. Information Technologies 5, 25–31 (2006) (in Russian)
19. Pertsovsky, S.L.: Building CAD-Systems for Modern Solo Dance. An Overview. Technical Report, Institute for Automation & Control Processes, FEBRAS (2006) (in Russian)
20. Pertsovsky, S.L., Varnina, A.S.: Development of the Intellectual CAD-System for Modern Solo Dance Based on Ontologies. Bulletin of FEBRAS 3, 163–169 (2006) (in Russian)
21. Kuzin-Alexinsky, A.S.: A Generator of Variations Using a Given Musical Theme. Informatics and Control Systems 1, 107–116 (2004) (in Russian)
22. Kleshchev, A.S., Gribova, V.V.: From an Ontology-Oriented Approach Conception to User Interface Development. Int. J. Information Theories & Applications 10(1), 87–93 (2003)
23. Kryukov, V.V., Shakhgelgyan, I.: Corporative Information Environment of a University. Dalnauka, Vladivostok (2007) (in Russian)
24. Michie, D.: Expert systems. Computer Journal 23(4), 369–376 (1980)
25. Kleshchev, A.S.: Tasks of inductive forming verbalizable knowledge in terms of ontologies. Scientific and Technical Information SeriesM2. 8, 8–18 (2003) (in Russian)
26. Artemjeva, I.L., Gavrilova, T.L., Gribova, V.V., et al.: The Multidiscipline Control System for Information Resources of Various Generality Levels. Control Sciences 4, 64–68 (2006) (in Russian)

27. Kleshchev, A.S., Orlov, V.A.: Computer Knowledge Banks. The Multipurpose Knowledge Bank. Information Technologies 2, 2–8 (2006) (in Russian)
28. Tyugu, E.H.: Conceptual programming. Nauka, Moscow (1984) (in Russian)
29. Babaev, I.O., Novikov, F.A., Petrushina, T.I.: Descartes Language – the Source Language of SPORA System. Applied Informatics, fasc. 1, 35–73 (1981) (in Russian)
30. Knyazeva, M.A., Kleshchev, A.S.: A Web-System for Computer Experiments in the Field of Program Transformations. Int. J. Information Theories & Applications 13(4), 331–336 (2006)

A Comparison of Content-Based Tag Recommendations in Folksonomy Systems

Jens Illig[1], Andreas Hotho[1], Robert Jäschke[1,2], and Gerd Stumme[1,2]

[1] Knowledge & Data Engineering Group, Department of Mathematics and Computer Science, University of Kassel, Wilhelmshöher Allee 73, 34121 Kassel, Germany
http://www.kde.cs.uni-kassel.de/
[2] Research Center L3S, Appelstraße 9a, 30167 Hannover, Germany
http://www.l3s.de/

Abstract. Recommendation algorithms and multi-class classifiers can support users of social bookmarking systems in assigning tags to their bookmarks. Content based recommenders are the usual approach for facing the cold start problem, i. e., when a bookmark is uploaded for the first time and no information from other users can be exploited. In this paper, we evaluate several recommendation algorithms in a cold-start scenario on a large real-world dataset.

1 Introduction

Social bookmarking systems allow web surfers to store and manage their bookmarks on a central server and not as usual within the browser. Thus, they allow their users to access bookmarks simultaneously from different computers and to share them with other users. The users have the possibility to assign freely chosen keywords, so-called *tags* to each resource, which can be used to structure and retrieve the stored bookmarks. To support the users in tagging, different types of recommendation algorithms are typically utilized by bookmarking systems.

The recommendation of tags can also be considered as a set of classification problems, since we can consider each tag as the name of a class. The number of classes is typically very high as folksonomy users are allowed to choose from as many different tags as they like. Since they typically assign more than one tag to a resource, this is a *multi-label classification* problem [25]. If the individual classifications are ranked by their confidence values such that those classifications occur first the classifier is most sure about, as we have done in this paper, the problem also has a ranking character.

There are two typical approaches to the recommendation problem: content-based approaches and collaborative filtering approaches [3]. While the former rely solely on the content of the documents, the latter take into account the behavior of similar users. Social bookmarking systems are an ideal scenario for the collaborative filtering approach, as the similarity of users can be measured by comparing their tagging behavior. Nevertheless, the so-called *cold start problem* also occurs in social bookmarking systems: When a resource is tagged for the first time by some user, no other user yields any recommendation about which tags to use for that particular resource. Therefore, content-based recommendations also have their use in social bookmarking systems.

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 136–149, 2011.

In this paper, we study different content-based recommenders and compare them on a real-world dataset – a crawl of the Delicious bookmarking system.[1] The main contribution is a comparison of state of the art tag recommenders, the adaptation of classifiers to this problem and a demonstration that content based recommenders are able to generalize and to make predictions for new web pages. The paper complements our work on collaborative filtering approaches [14]. A more detailed discussion of its findings can be found in the bachelor thesis [13] of Jens Illig.

The paper is organized as follows. In Section 2, we introduce folksonomies, the underlying data structure of social bookmarking systems. In Section 3, we discuss related work. Section 4 defines the problem and describes the classifiers that we used. Section 5 describes the data set that we used, and the preprocessing that we performed. We discuss our findings in Section 6 and future work in Section 7.

2 Social Resource Sharing and Folksonomies

The central data structure of a social bookmarking system is a *folksonomy*. It consists of the assignments of tags to resources by some users. The following definition, taken from [12], formalizes this idea:[2]

Definition 1 (Folksonomy). *A folksonomy is a tuple* $\mathbb{F} := (U, T, R, Y)$ *where*

- U *is a finite set of users,*
- T *is a finite set of tags,*
- R *is a finite set of resources, and*
- $Y \subseteq U \times T \times R$ *is a ternary relation between users, tags, and resources. An element* (u, t, r) *of* Y *is called a* tag assignment (TAS) *and represents the fact that user* u *has assigned tag* t *to resource* r*.*

The set of tags that user u *has assigned to resource* r *is given by* $T_{ur} := \{t \in T \mid (u, t, r) \in Y\}$*. If* T_{ur} *is non-empty, then we call the tuple* (u, T_{ur}, r) *the* post *of user* u *for resource* r*.*

Note that the set T of tags may grow over time, as there are no pre-defined keywords – the user is free to come up with arbitrary new tags. A resource is usually labeled by multiple users and tags may be assigned multiple times to the same resource by different users.

For content-based recommendations, we will abstract from the user dimension. Therefore, we introduce the set of *binary tag assignments (BTAS)* as projection I of Y on the tag and resource dimensions: $I := \{(t, r) \in T \times R \mid \exists u \in U \colon (u, t, r) \in Y)\}$. If $T_r := \{t \in T \mid (t, r) \in I\}$ is non-empty, then we call the tuple (T_r, r) the *bpost* for resource r. We use this projection, because our content based recommendation approach is based on global inter-user classifiers instead of individual classifiers for each user and therefore outputs no user information. Individual classification for each user would require very active users to gain enough data for classifier trainig.

[1] `http://delicious.com/`

[2] In [12], we have additionally introduced a user-specific sub-tag/super-tag relation, which we will ignore for the purpose of this paper.

3 Related Work

General overviews on the rather young area of folksonomy systems and their strengths and weaknesses are given in [11,18,19]. In [20], Mika defines a model of semantic-social networks for extracting lightweight ontologies from Delicious. Recently, work on more specialized topics, such as structure mining on folksonomies – e. g., to visualize trends [8] and patterns [23] in users' tagging behavior – as well as ranking of folksonomy contents [12], analyzing the semiotic dynamics of the tagging vocabulary [5], or the dynamics and semantics [10] have been presented.

The literature concerning the problem of tag recommendations in folksonomies is still sparse. The existent approaches usually lie in the collaborative filtering and information retrieval areas. In [21], [4], and [14], algorithms for tag recommendations are devised based on content-based filtering techniques. Xu et al. [29] introduce a collaborative tag suggestion approach based on the HITS algorithm [16]. A goodness measure for tags, derived from collective user authorities, is iteratively adjusted by a reward-penalty algorithm. Benz et al. [2] introduce a collaborative approach for bookmark classification based on a combination of nearest-neighbor-classifiers. There, a keyword recommender plays the role of a collaborative tag recommender, but it is just a component of the overall algorithm, and therefore there is no information about its effectiveness alone. Basile et al. [1] suggest an architecture for an intelligent recommender tag system. In [9,28,27], the problem of tag-aware resource recommendations is investigated. The standard tag recommenders, in practice, are services that provide the most-popular tags used for a particular resource. This is usually done by means of tag clouds where the most frequently used tags are depicted in a larger font or otherwise emphasized.

First work which utilized machine learning algorithms to predict tags based on the content is reported in [25]. The reported results for four real world dataset are very promising but limited to only two models, a new gaussian process and an SVM model. Results for a vector space model and a poisson mixture model are reported in [26]. The results are similar to those we report here for other machine learning methods.

Most recently, the ECML PKDD 2009 Discovery Challenge[3] has addressed the problem of tag recommendations in folksonomies. Most of the proposed approaches rely on a combination of good preprocessing, some external knowledge sources and a good heuristic to choose the best set of tags.

4 Tag Recommendations as Text Classification Problem

4.1 Definition of the Problem

In [14], we have studied tag recommendations based on a collaborative filtering approach. But in a dynamic setting, such as our web bookmarking scenario, new web pages show up frequently. When a new page is bookmarked for the first time, the only information about it is its full text. Our aim is to learn tag recommendations that are based on this information.

We formalize the problem as follows. Let $\mathbb{F} := (U, T, R, Y)$ be a folksonomy, where the set R of resources consists of web pages. The web pages are modeled by the

[3] `http://www.kde.cs.uni-kassel.de/ws/dc09/`

bag-of-words approach, i. e., a mapping $\mathrm{vec}\colon R \to \mathbb{R}^V$, where V is the set of all[4] words occurring in at least one document, and where $\mathrm{vec}(r)_v$ is the number of occurrences of word v on web page r. We applied weighting of term frequencies by their inverse document frequency (in that combination abbreviated as *tf-idf*) to that mapping.[5]

For the evaluation, we assume that the folksonomy $\mathbb{F}$ is split into a training and a test set, i. e., into:[6]

$$\mathbb{F}_{\mathrm{train}} = (U_{\mathrm{train}}, T_{\mathrm{train}}, R_{\mathrm{train}}, Y_{\mathrm{train}}) \quad \text{and} \quad \mathbb{F}_{\mathrm{test}} = (U_{\mathrm{test}}, T_{\mathrm{test}}, R_{\mathrm{test}}, Y_{\mathrm{test}})$$

The *problem of learning tag recommendations* consists in finding, based on the information in $\mathbb{F}_{\mathrm{train}}$ and some $n \in \mathbb{N}$, a function $\varphi_n\colon \mathbb{R}^V \to \mathfrak{P}_n(T)$,[7] such that, for all resources r in R_{test}, $\varphi_n(\mathrm{vec}(r))$ is a good approximation for the tags of r. As usual, we will measure the quality of the approximation with precision and recall, see Section 6.1.

4.2 Classifiers

In order to solve the problem of finding a concrete mapping φ_n, we applied different machine learning algorithms which are suitable for the text classification task (cf. [24]). In the experiments, we compared the following models: Support Vector Machines (SVM), multinominal naïve Bayes, Rocchio, k-Nearest-Neighbor (k-NN), and – as a simple baseline – the most popular tags for the document. At the end, all models provide a function $\breve{\Phi}_t^{\neg t}\colon \mathbb{R}^V \to \mathbb{R}$ which returns, for $\vec{x} \in \mathbb{R}^V$, a confidence value $\breve{\Phi}_t^{\neg t}(\vec{x})$ describing how confident the model is in assigning tag t to a resource $r \in R$ with $\mathrm{vec}(r) = \vec{x}$. The recommendation $\varphi_n(r)$ then consists of those n tags $t \in T$ having the highest values $\breve{\Phi}_t^{\neg t}(\mathrm{vec}(r))$.

The functions $\breve{\Phi}_t^{\neg t}$ are either computed directly – this approach is called t-vs-$\neg t$ or one-vs-all (abbreviated as *1vsAll*) – or calculated from multiple confidence values of pairwise tag comparisons $\breve{\Phi}_x^y$ where $\breve{\Phi}_x^y(\mathrm{vec}(r))$ is the confidence in the decision to prefer tag x instead of tag y for resource r. The latter approach is called one-vs-one. For all learning algorithms except k-Nearest-Neighbor where only one-vs-all has been applied, we experimented both with one-vs-all and one-vs-one.

For one-vs-one, we evaluated two different variants for calculating a single confidence function $\breve{\Phi}_t^{\neg t}$ from all confidence functions $\{\breve{\Phi}_x^y \mid x \in T_{\mathrm{train}} \wedge y \in T_{\mathrm{train}} \wedge x \neq y \wedge (x = t \vee y = t)\}$. The first uses simple Boolean vote adding (abbreviated as *1vs1bool*) and requires hard classifications for every tag-vs-tag pair to increase a vote counter for the winning tag of the pair. A confidence threshold of zero has been used to get this hard classification which is motivated by the fact that most of the tested classifiers are directed to output confidence values with positive or negative values for indicating preference of tag in favor of $\neg tag$ respectively tag x in favor of tag y. A confidence value exactly equal to zero leads to no vote for any of the two tags in the one-vs-one pair.

[4] In this paper, we did not apply stopword removal.

[5] We also made the same classification experiments without such weighting but the best results of every classifier family were achieved with tf-idf.

[6] The specific splitting approach that we used for this paper is described in Section 5.2.

[7] $\mathfrak{P}_n(T)$ stands for the set of all subsets of T with exactly n elements.

The other tested variant of defining $\breve{\Phi}_t^{\neg t}$ uses confidence adding (abbreviated as *1vs1conf*):

$$\breve{\Phi}_t^{\neg t}\colon \mathbb{R}^V \to \mathbb{R};\quad \vec{x} \mapsto \sum_{t' \in T_{\text{train}} \setminus \{t\}} \breve{\Phi}_t^{t'}(\vec{x}) - \breve{\Phi}_{t'}^{t}(\vec{x}) \tag{1}$$

All presented algorithms follow the same principle for computing the functions $\breve{\Phi}_t^{\neg t}$ in case of one-vs-all and the functions $\breve{\Phi}_x^y$ in the one-vs-one case: Let 0 and 1 stand for $\neg t$ and t, resp., in the first case, and for tag x and tag y, resp., in the second case. Furthermore, let $\mathbb{V}_{\text{train}} = \{\text{vec}(r) \mid r \in R_{\text{train}}\}$ be the set of training feature vectors and $\Phi_1^0\colon \mathbb{V}_{\text{train}} \to \{0,1\}$ be a function represesenting some mapping of known information – or decisions – about these training examples that is to be learned. Then each machine learning algorithm finds a function $\breve{\Phi}_1^0\colon \mathbb{R}^V \to \mathbb{R}$ which maps to real valued confidence values indicating how much more suitable decision 1 is in favor of decision 0 for a feature vector in $\mathbb{R}^V$ regarding an internal model that is learned from the training examples.

SVM. Support Vector Machines are classifiers that separate the feature hyperspace of some dimension $|V|$ into two subspaces divided by a $|V|-1$ dimensional hyperplane. Thereby SVMs also try to find a hyperplane position that provides a broad 'safety' space around the hyperplane instead of simply focussing on a small training error rate.

As used for example in [22], two parameters, $C^+ \in \mathbb{R}$ and $C^- \in \mathbb{R}$ define the relative importance of consistency with positive and negative training examples against safety space maximization. For the experiments with the SVM machine learning method, a marginally modified implementation of the linear C-SVM algorithm from the library libSVM [7] has been used that outputs its internal hyperplane distance as confidence values instead of hard classifications. We experimented both with the default setting $C = C^+ = C^- = 1$ and a second variant using

$$C^- = \frac{|\{r \in R_{\text{train}} \mid \Phi_1^0(\text{vec}(r)) = 0\}|}{2 \cdot |\{r \in R_{\text{train}} \mid \Phi_1^0(\text{vec}(r)) = 1\}|} \quad \text{together with} \quad C^+ = 2 \cdot (C^-)^2$$

This asymmetric setting (which is marked as $C = +/-$ in the evaluation section) is motivated by the observation that a negative resource/tag example can either be a 'real' negative example (i. e., the tag indeed does not fit to the resource), or a 'missed' positive example (i. e., the tag semantically belongs to the resource, but has not yet been assigned explicitly to it by any of the users of the system). Thus, the cost of misclassifying a positive training example (C^+) should be higher than the cost of misclassifying a negative example. However, setting C^+ too high in relation to C^- may lead to a trivial positive classifier. The above given settings of C^- and C^+ have been determined on the basis of multiple small manually constructed two-dimensional test datasets. Experiments have been conducted with and without scaling all document feature vectors to an Euclidean length of one before training and classification (denoted by *lnorm* and *nolnorm*, resp., in the evaluation section).

Multinomial Naïve Bayes. This method applied to tag classification calculates a probability estimate $P(t|r)$ for the observation of tag t given an observation of a resource r.

We calculated the log odds ratio of probabilities from a multinomial model with document model based parameter estimation as described in [15], which leads to[8]

$$\breve{\Phi}_t^{\neg t}(\mathrm{vec}(r)) = \log\left(\frac{P(t|r)}{P(\neg t|r)}\right) = \log\left(\prod_{v\in r}\left(\frac{P(v|t)}{P(v|\neg t)}\right)^{\mathrm{vec}(r)_v}\cdot\frac{P(t)}{P(\neg t)}\right) \quad (2)$$

$$\text{with} \quad P(v|t) = \sum_{r'\in R_{\text{train}}} P(r'|t)\cdot P(v|r') \ . \quad (3)$$

We estimated $P(r'|t)$, $P(v|r')$ and $P(t)$ as well as $\neg t$ variants directly from the relative TAS and term occurrence frequencies in the training corpus. To avoid $P(v|t) = 0$ as a factor in the right term of Equation 2, a virtual post $p^\star = \{u^\star\} \times T_{\text{train}} \times \{r^\star\}$ has been added to the training dataset. $r^\star$ is made up of one occurrence of every feature known from the training dataset plus a virtual wildcard feature. During classification, each new feature not known from the training dataset has been treated equally to the wildcard feature.

Rocchio. This centroid based method builds class representation vectors that are compared to resource representation vectors in order to find some similarity measure as the confidence output value. As presented for example in [24], we calculated positive and negative centroid vectors for the training classes 0 and 1 as follows

$$\vec{c}^{\,1} = \frac{1}{|R^1_{\text{train}}|}\sum_{r_{\text{train}}\in R^1_{\text{train}}}\mathrm{vec}(r_{\text{train}}) \qquad \vec{c}^{\,0} = \frac{1}{|R^0_{\text{train}}|}\sum_{r_{\text{train}}\in R^0_{\text{train}}}\mathrm{vec}(r_{\text{train}})$$

With these centroids we define $\breve{\Phi}$ as follows

$$\breve{\Phi}^0_1(\mathrm{vec}(r)) = \cos\left(\measuredangle\left(\beta\frac{\vec{c}^{\,1}}{\|\vec{c}^{\,1}\|} - \gamma\frac{\vec{c}^{\,0}}{\|\vec{c}^{\,0}\|}, \mathrm{vec}(r)\right)\right)$$

Classifier setups have been evaluated with $\beta = 1$ in combination with both $\gamma = 0$ and $\gamma = 1$. Additionally, we experimented with TAS weighted centroids, but yielded slightly lower effectiveness. Furthermore, our experiments with Euclidean distance always led to effectiveness below the baseline.

***k*-NN.** We have run the k-Nearest-Neighbor method considering the 30 nearest neighbor documents and using a confidence calculation scheme taken from [24], that is

$$\breve{\Phi}^0_1(\mathrm{vec}(r)) = \sum_{r_{\text{train}}\in MostSim_k} f_{\text{sim}}\left(\mathrm{vec}(r), \mathrm{vec}(r_{\text{train}})\right)\cdot\Theta^0_1(r_{\text{train}})$$

$$\Theta^0_1(r_{\text{train}}) = \begin{cases} 1 & \text{, if } \Phi^0_1(\mathrm{vec}(r_{\text{train}})) = 1 \\ 0 & \text{, otherwise} \end{cases}$$

Here, $MostSim_k \subseteq R_{\text{train}}$ is the set of those training instances that are among the k most similar instances compared by similarity measure $\mathrm{sim} : \mathbb{R}^V \times \mathbb{R}^V \to \mathbb{R}$ to the argument instance r which is to be classified. We used $\mathrm{sim}(x,y) = \cos\left(\measuredangle(x,y)\right)$. For the similarity weighting function f_{sim}, both $f_{\text{sim}}(x,y) = 1$ and $f_{\text{sim}}(x,y) = \sim(x,y)$

[8] We use $v \in r$ here for $v \in \{v' \mid v' \in V \wedge \mathrm{vec}(r)_{v'} > 0\}$.

have been evaluated. Additionally, we experimented with an alternative definition of $\Theta_1^0(r_{\text{train}})$ which also takes into account how many users assigned a tag to a resource in the training set:

$$\Theta_t^{\neg t}(r_{\text{train}}) = \begin{cases} \log\left(|\{u \in U_{\text{train}} \mid (u, t, r_{\text{train}}) \in Y_{\text{train}}\}| + 2\right), \text{ if } \Phi_t^{\neg t} = t, \\ -\log\left(|TAS_{\text{train}}^{\neg t}(r_{\text{train}})|\right), \text{ if } \Phi_t^{\neg t} = \neg t \text{ and } |TAS_{\text{train}}^{\neg t}(r_{\text{train}})| \geq 1 \\ 0, \text{otherwise} \end{cases}$$

where $TAS_{\text{train}}^{\neg t}(r_{\text{train}}) = \{(u, t') \in U_{\text{train}} \times (T_{\text{train}} \setminus \{t\}) \mid (u, t', r_{\text{train}}) \in Y_{\text{train}}\}$. The logarithm is used for damping and $+2$ is used to slightly linearize the logarithm in order to weight positive neighbors strongly even with few TAS.

5 Evaluation Setting

The dataset used for the experiments is a crawl of the social bookmarking system Delicious downloaded between 2005-07-27 and 2005-07-30 [12]. It consists of $75,242$ users, $533,191$ tags and $3,158,297$ resources, related by in total $17,362,212$ tag assignments. The full text of all $3,158,297$ resources has also been downloaded in 2005. Unfortunately, the protocol response headers of the resource downloads were lost. For that reason it was at first unclear how many resources were error code pages and which resources were correctly transferred resources. There was also no information about the MIME type of the resources, the encoding, or the language in case of text resources.

5.1 Preprocessing

Based on MIME type detections of a magic byte sequence algorithm,[9] all resources for which the detected MIME type neither started with "text" nor contained the substring "html" have been filtered out. In order to escape all character set problematics, the document corpus has been restricted to 7 bit ASCII encoded documents.[10] All pages estimated being erroneous in terms of a non-HTTP 2xx response have been pruned. Since the HTTP response status codes had not been captured during the download and error pages usually contain customized explanations for human readers, we identified such documents by an SVM trained by a set of $1,271$ successfully and $1,000$ unsuccessfully re-crawled resources. A ten fold cross-validation of this classifier showed an eleven point average precision of 0.96.

Primarily because of the tokenization problem with natural text of some languages, but also with respect to the comparability of possible future stemming experiments on the same dataset, only English text documents were evaluated. Language guessing was done by making use of the n-gram method [6].[11] Whenever documents contained explicit information about their language, we doubled the score of that language.

[9] We used the algorithm from the the "data and metadata getting" Java framework *Aperture* (`http://aperture.sourceforge.net/`) in version 1.0.1-beta.

[10] Detected by application of the *jchardet* (`http://jchardet.sourceforge.net/`) library, which is a Java port of the Mozilla universal charset detector [17].

[11] Applied via the character based part of the *ngramj* library (`http://ngramj.sourceforge.net/`).

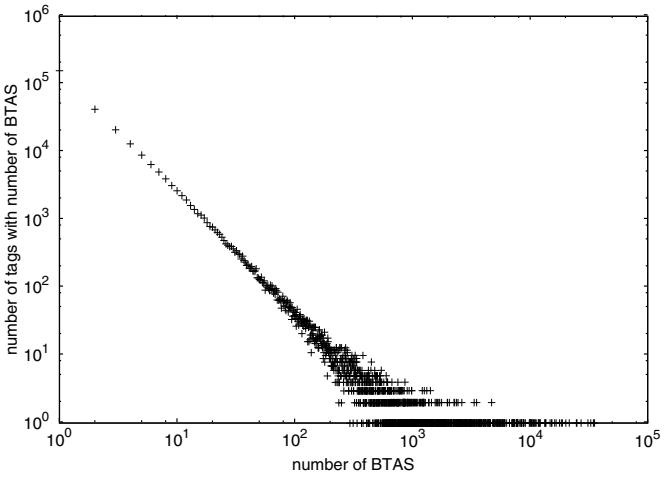


Fig. 1. Tag frequencies for all resources. The fact that the data points almost form a line hints at the presence of a so-called *power law* distribution, which is typical for many human-driven activities.

The pruning steps (pruning of error pages, non-text/html text, non-English documents and non-7-bit ASCII) reduced our initial folksonomy. We removed then all users and all resources that were no longer related to a resource. We obtained in total $65,177$ users, $299,305$ tags, and $1,113,405$ resources.

Figure 1 shows the frequency distribution of the tags. Each cross is representing one tag. The right-most cross (which is located on the x-axis) says that there is exactly one ($= 10^0$) tag that occurs in $35,307$ BTAS, while the left-most cross (which is located on the y-axis) says that there are $158,183$ different tags that occur in only one BTAS each. Since such rare tags are very difficult to predict, and since we had a variety of algorithms and parameter settings that we wanted to evaluate, we had to reduce the data further. Hence, we restricted the set of tags to the 15 most popular tags, in order to reduce the complexity of the learning problem.[12] The remaining folksonomy $\mathbb{F}$ consists of $65,177$ users, 15 tags, and $1,113,405$ resources. Users and resources that are not related to any of the 15 most frequent tags have not been deleted, as they were used as negative training data.

The remaining preprocessing steps generate the vector space representation $\mathrm{vec}\colon R \to \mathbb{R}^V$ of the full text of the web pages. For (X)HTML documents, a parser has been used that passes through all non-markup as long as it is located inside of one of the HTML-tags `head` or `body` and outside of all the HTML-tags `embed`, `object`, `style`, `applet`, and `script`. The parser has been configured to filter out documents containing the `frameset` HTML-tag.

Two types of document features have been extracted from text documents during the build of *bag-of-words* feature vector representations. The first type of features is

[12] Faced with the limitation of our computing machinery (an Opteron PC with 8×2 GHz and 32 GB main memory), we had to decide whether to include more different algorithms, or to run a more extensive comparison of fewer algorithms. We decided to go for the former.

tokenized text with terms and character sequences. All tokens were changed to lowercase before frequency counting. With respect to the other feature type, occurrences of the HTML-tags `img, a, code, p, object, applet, embed, form, cite, dfn, q, samp` have been counted separately as features. We did not apply stopword removal.

Then the vector space representation of the documents has been built, as described in Section 4.1.

5.2 Training and Test Datasets

We follow the typical evaluation setting for supervised learning tasks by splitting the available data, i. e., the folksonomy $\mathbb{F}$ that resulted from the preprocessing as described above, into a training and a test data set. The split is based on the date of the posts. All posts between 2003-10-01 and 2004-08-26 have been used for the training dataset $\mathbb{F}_{\text{train}}$, resulting in $4,236$ users,

For the set of test documents, we considered all $10,602$ documents that occurred in posts between 2004-08-27 and the end of 2004-09-05. From these, we removed all $2,417$ documents which also occurred in posts from the training set. By removing all documents that are in both the training and the test dataset, we avoid the problem of evaluating our approach on already seen data, which would bias the evaluation. All TAS after 2004-08-27 (including those after 2004-09-05) referring to the remaining $8,185$ test documents have been used to find a testset of BTAS. (By not limiting to posts before 2004-09-05, we extend the set of BTAS and can thus use the maximal available information for the evaluation.) From the remaining documents, we removed all documents that were not tagged by any of the 15 most frequent tags. As an additional attempt to reduce the problem of suitable recommended tags that have not been assigned in our dataset, we limited the set to only those documents with at least ten TAS in the whole dataset. Again we removed unconnected users. The resulting test set $\mathbb{F}_{\text{test}}$ contains $40,632$ users,15 tags, and $1,926$ resources.

6 Experiments

6.1 Evaluation Settings

We evaluated all the recommenders on the test dataset $\mathbb{F}_{\text{test}}$. Since all recommendations were non-personalised, we removed the user dimension of $\mathbb{F}_{\text{test}}$ – i. e., we considered the set $I_{\text{test}} \subseteq T_{\text{test}} \times R_{\text{test}}$ of BTAS only.

For evaluating a recommender φ for each resource r in R_{test} and i between 1 and 5, we computed a recommendation $\varphi_i(r)$, recommending between one and five tags. For each of these combinations, precision and recall were computed:

$$\text{precision}(\varphi_i, r) = \frac{|\varphi_i(r) \cap T_r|}{|\varphi_i(r)|} \qquad \text{recall}(\varphi_i, r) = \frac{|\varphi_i(r) \cap T_r|}{|T_r|}$$

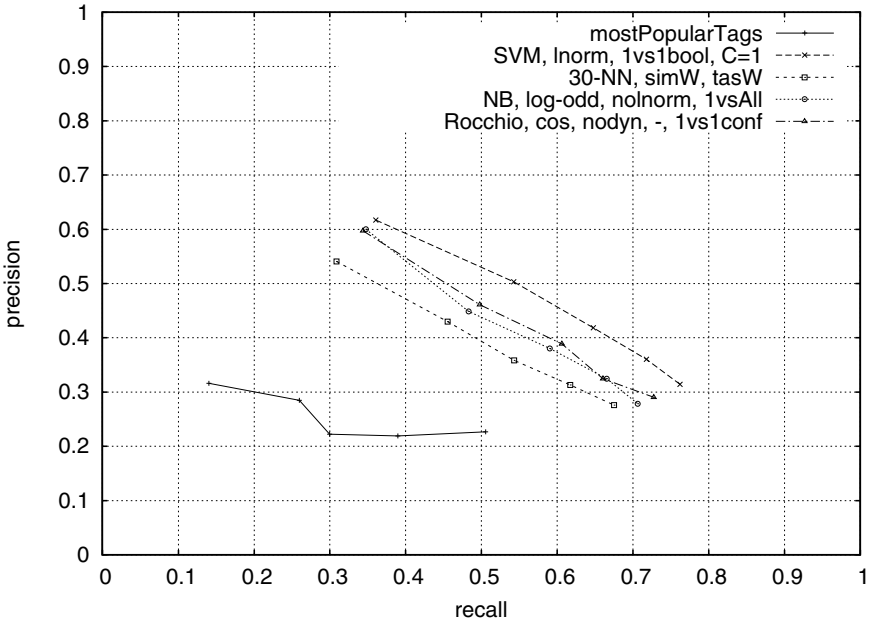


Fig. 2. Precision and recall of the best recommenders of different classifier types, averaged over all test documents that have at least 10 TAS in the whole dataset. All use tf-idf values.

For each recommender and for each $i = 1, \ldots, 5$, we averaged precision and recall over all resources in R_{test}:

$$\text{precision}(\varphi_i) = \frac{1}{|R_{\text{test}}|} \sum_{r \in R_{\text{test}}} \text{precision}(\varphi_i, r)$$

$$\text{recall}(\varphi_i) = \frac{1}{|R_{\text{test}}|} \sum_{r \in R_{\text{test}}} \text{recall}(\varphi_i, r)$$

6.2 Comparison of the Classifiers

All classifiers were evaluated with several parameter settings, which, due to space restrictions, cannot all be presented. For more details, see the bachelor thesis [13] of Jens Illig.

An overall comparison of all approaches is shown in Figure 2. For the sake of readability, we did not display all parameter settings, but only one or two of those that performed best for each classifier type. In the diagram, one can see that the SVM with one-vs-one learning is clearly more effective than the other classifiers. One-vs-one is also the best choice for the Rocchio method. Thereby, a confidence adding variant without TAS weighted centroids turned out to be most effective. However, our experiments showed that the worst cosine-based Rocchio classifier is only about 0.04 precision score points less effective at similar recall levels. Most clearly, those variants with $\gamma = 0$

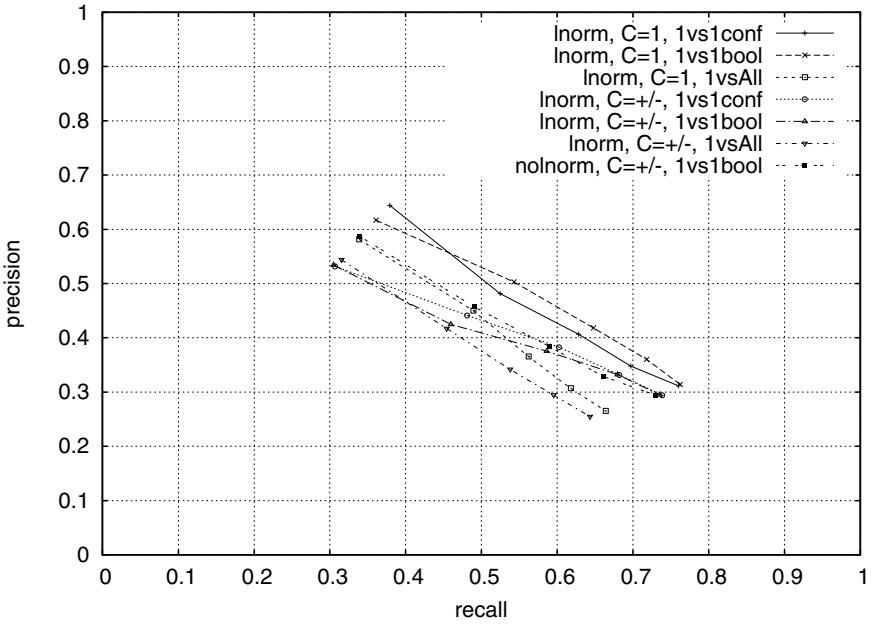


Fig. 3. Tf-idf precision and recall for different parameter settings of the SVM, averaged over all test documents not occuring in the training set that have at least 10 TAS in the whole dataset

were the least effective among them. Another well functioning classifier is log-odds ratio multinomial Naïve Bayes. For that classifier type, length normalization has turned out to be counterproductive for tag recommendation. 30-NN is clearly less effective. The best 30-NN variants use our TAS weighting scheme, which seems to increase precision by ca. 0.04. Similarity weighting also slightly increased precision. As expected, simple most popular tag recommendation is less effective than almost all content based methods – only the highly ineffective Rocchio methods with Euclidean distance (not displayed in Figure 2) are worse.

As the SVM performs best we present results of it comparing different parameter settings. Figure 3 shows the results for the most interesting settings of the SVM. The five recommendations $\varphi_1, \ldots, \varphi_5$ of each setting are plotted together in one curve. The left-most node of each curve represents the one-element-recommendation φ_1, while the right-most node represents the five-element-recommendation φ_5. As one can see, all curves are monotonically decreasing. This shows that recall is growing for all recommenders with an increasing number of recommendations while precision is falling.

The figure shows that the best settings for the SVM are those with $C = 1$ parameterization. A possible explanation is that our dynamically calculated parameterization (called "$C + /-$" Figure 3) with its higher C values tends to overfit. This is supported by our observation that in another evaluation that is based solely on repeatedly posted documents, these classifiers show higher effectiveness than the corresponding $C = 1$ variants. With the better working $C = 1$ SVM configuration, Figure 3 also shows that the Boolean adding one-vs-one variant is most effective at higher recall levels, while,

with confidence adding, the first item can be recommended more precisely. This might be explained by real-valued confidence values of the confidence adding classifier variant where, in contrast, Boolean adding uses only integer vote counts as confidence values, which, as we observed, often leads to many tag suggestions with equal confidence output so that these cannot be ordered any further. Without length-normalization, SVM effectiveness is in most cases lower, especially for one-vs-all classifiers. The best non length-normalized variant is Boolean adding one-vs-one with "$C+/-$", but it only has around 0.02 more precision than the $C = 1$ variant at similar recall levels.

7 Conclusion and Outlook

In this paper, we evaluated the effectiveness of multiple text classification methods and variants applied to a scenario that is compatible with the common text classification evaluation practice of disjoint training and test scenarios but still represents a realistic and pure cold start tag recommender evaluation scenario. Thereby, we identified a problem in the open world characteristic of the dataset and developed an evaluation scheme that addresses it.

Some algorithms have been slightly modified in various ways to make use of tag assignment frequencies by multiple users. Improvements by these extensions have been detected for the case of a TAS weighted 30-Nearest-Neighbors algorithm. Nevertheless, we found that a one-vs-one SVM variant on length normalized document feature vectors is the most effective of all evaluated classifiers. We could show that folksonomy tag assignments can be learned by application of machine learning techniques to address the cold start problem of collaborative recommender systems.

In the future, our experiments can be extended to other classifier algorithms, like, for example, boosting, decision trees, and rule based learners. Also transductive approaches seem promising in terms of the open world problem. Other possible extensions include stemming, term space reduction, different feature reweighting methods, and classification of documents in multiple languages.

Another open task is to evaluate and compare the effectiveness of content based and collaborative approaches (on a test set of already posted resources). The next step is then to develop combined approaches that rely on both the high effectiveness of collaborative methods on documents with known tag assignments and the strengths of content based approaches to overcome the cold start problem.

Acknowledgement. Part of this research was funded by the European Commission in the projects Nepomuk (FP6–027705) and TAGora (FP6–IST5–34721).

References

1. Basile, P., Gendarmi, D., Lanubile, F., Semeraro, G.: Recommending smart tags in a social bookmarking system. In: Bridging the Gap between Semantic Web and Web 2.0 (SemNet 2007), pp. 22–29 (2007)
2. Benz, D., Tso, K., Schmidt-Thieme, L.: Automatic bookmark classification: A collaborative approach. In: Proceedings of the Second Workshop on Innovations in Web Infrastructure (IWI 2006), Edinburgh, Scotland (2006)

3. Burke, R.: Hybrid recommender systems, survey and experiments. User Modeling and User Adapted Interaction 12(4), 331–370 (2002)
4. Byde, A., Wan, H., Cayzer, S.: Personalized tag recommendations via tagging and content-based similarity metrics. In: Proceedings of the International Conference on Weblogs and Social Media, Boulder, Colorado, USA (March 2007)
5. Cattuto, C., Loreto, V., Pietronero, L.: Collaborative tagging and semiotic dynamics (May 2006), http://arxiv.org/abs/cs/0605015
6. Cavnar, W.B., Trenkle, J.M.: N-gram-based text categorization. In: Proceedings of SDAIR 1994, 3rd Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, US, pp. 161–175 (1994)
7. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines (2001), Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
8. Dubinko, M., Kumar, R., Magnani, J., Novak, J., Raghavan, P., Tomkins, A.: Visualizing tags over time. In: Proc. of the 15th International WWW Conference, Edinburgh, Scotland (2006)
9. Firan, C.S., Nejdl, W., Paiu, R.: The benefit of using tag-based profiles. In: 5th Latin American Web Congress, Santiago de Chile, 31 October - 2 November (2007)
10. Halpin, H., Robu, V., Shepard, H.: The dynamics and semantics of collaborative tagging. In: Proceedings of the 1st Semantic Authoring and Annotation Workshop (SAAW 2006), Atlanta, Georgia, USA (2006)
11. Hammond, T., Hannay, T., Lund, B., Scott, J.: Social Bookmarking Tools (I): A General Review. D-Lib Magazine 11(4) (April 2005)
12. Hotho, A., Jäschke, R., Schmitz, C., Stumme, G.: Information Retrieval in Folksonomies: Search and Ranking. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 411–426. Springer, Heidelberg (2006)
13. Illig, J.: Machine learnability analysis of textclassifications in a social bookmarking folksonomy. Bachelor thesis. University of Kassel, Kassel (2008)
14. Jäschke, R., Marinho, L., Hotho, A., Schmidt-Thieme, L., Stumme, G.: Tag recommendations in social bookmarking systems. AI Communications 21(4), 231–247 (2008)
15. Kim, S., Rim, H., Yook, D., Lim, H.: Effective methods for improving naive bayes text classifiers (2002)
16. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. Journal of the ACM 46(5), 604–632 (1999)
17. Li, S., Momoi, K.: A composite approach to language/encoding detection. In: 19th International Unicode Conference, San Jose, California, USA (2001)
18. Lund, B., Hammond, T., Flack, M., Hannay, T.: Social Bookmarking Tools (II): A Case Study - Connotea. D-Lib Magazine 11(4) (April 2005)
19. Mathes, A.: Folksonomies – Cooperative Classification and Communication Through Shared Metadata (December 2004), http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html
20. Mika, P.: Ontologies Are Us: A Unified Model of Social Networks and Semantics. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 522–536. Springer, Heidelberg (2005)
21. Mishne, G.: Autotag: a collaborative approach to automated tag assignment for weblog posts. In: WWW 2006: Proceedings of the 15th International Conference on World Wide Web, pp. 953–954. ACM Press, New York (2006)
22. Morik, K., Brockhausen, P., Joachims, T.: Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In: Bratko, I., Dzeroski, S. (eds.) Proceedings of the 16th International Conference on Machine Learning (ICML 1999), Bled, Slovenia, June 27-30, pp. 268–277. Morgan-Kaufman Publishers, San Francisco (1999)

23. Schmitz, C., Hotho, A., Jäschke, R., Stumme, G.: Mining association rules in folksonomies. In: Batagelj, V., Bock, H.-H., Ferligoj, A., Žiberna, A. (eds.) Data Science and Classification: Proc. of the 10th IFCS Conf., Studies in Classification, Data Analysis, and Knowledge Organization, pp. 261–270. Springer, Heidelberg (2006)
24. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1), 1–47 (2002)
25. Song, Y., Zhang, L., Lee Giles, C.: A sparse gaussian processes classification framework for fast tag suggestions. In: CIKM 2008: Proceeding of the 17th ACM Conference on Information and Knowledge Mining, pp. 93–102. ACM, New York (2008)
26. Song, Y., Zhuang, Z., Li, H., Zhao, Q., Li, J., Lee, W.-C., Lee Giles, C.: Real-time automatic tag recommendation. In: SIGIR 2008: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 515–522. ACM, New York (2008)
27. Tso-Sutter, K., Marinho, L.B., Schmidt-Thieme, L.: Tag-aware recommender systems by fusion of collaborative filtering algorithms. In: Proceedings of 23rd Annual ACM Symposium on Applied Computing (SAC 2008), Edinburgh, Scotland (2007)
28. Xu, Y., Zhang, L., Liu, W.: Cubic analysis of social bookmarking for personalized recommendation. In: Zhou, X., Li, J., Shen, H.T., Kitsuregawa, M., Zhang, Y. (eds.) APWeb 2006. LNCS, vol. 3841, pp. 733–738. Springer, Heidelberg (2006)
29. Xu, Z., Fu, Y., Mao, J., Su, D.: Towards the semantic web: Collaborative tag suggestions. In: Proceedings of the Collaborative Web Tagging Workshop at the WWW 2006, Edinburgh, Scotland (2006)

Data Weeding Techniques Applied to Roget's Thesaurus

Uta Priss and L. John Old

Edinburgh Napier University, School of Computing
www.upriss.org.uk, j.old@napier.ac.uk

Abstract. It can be difficult to automatically generate "nice" graphical representations for concept lattices from lexical databases, such as Roget's Thesaurus, because the data sources tend to be large and complex. This paper discusses a variety of "data weeding" techniques that can be applied in order to reduce the size of a concept lattice, first in general, and then with respect to Roget's Thesaurus. The aim is that resulting lattices should display neither too much, nor too little information, independently of which search terms have been entered by a user.

1 Introduction

Large and complex concept lattices provide a challenge for Formal Concept Analysis (FCA) because they can be difficult to display and navigate by users. Lattices that are automatically derived from lexical databases, such as Roget's Thesaurus, tend to be large and complex. What is needed is some purpose-driven manner of size reduction and selection of subsets of the data. Decisions about which data to select resemble a form of filtering analogous to "weeding" in a garden, because whether a plant is considered useful or a "weed" does not usually depend on structures of the plant itself but solely on whether it is desired in a location. Thus we define "data weeding" techniques as techniques that select data from a given set based on the specific needs and purpose of an application.

A variety of existing Formal Concept Analysis methods can be called data weeding techniques. These techniques, for example, select subsets of the data or reduce the visual complexity of the concept lattice. The choice of the technique usually relies on the type of application. This paper provides an overview of data weeding techniques in general, and then analyses their applicability to Roget's Thesaurus. The goal for this research is to automatically generate lattices from Roget's Thesaurus in an on-line interface[1] in a manner that adjusts the data weeding techniques for each request.

Data weeding techniques for FCA can be categorised into four types as shown in Table 1. The first type, visual reduction techniques, are techniques that change how the data is displayed without changing the mathematical structure of the underlying concept lattices. The second type, faceting and plain scaling, are techniques which lead to a division of the original concept lattices into smaller lattices without information loss. The division should be meaningful with respect to the content of the lattice. For example, if the attributes can be naturally subdivided into a set of partitions, then a separate lattice can be drawn for each partition of the data. In FCA, this is called "plain"

[1] http://www.roget.org

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 150–163, 2011.

scaling (Ganter & Wille, 1999) for single-valued contexts. The third type, pruning and restricting, consists of techniques which reduce concept lattices by removal of objects, attributes or concepts usually based on statistics. The fourth type, decomposition and general scaling, decomposes a lattice and at the same time reduces complexity, for example, by aggregating objects or attributes. The third and fourth types usually cause some loss of information because of summarisation, abstraction or reduction.

Table 1. Four types of data weeding techniques

Type of reduction	Effect on lattice	Loss of information
Visual reduction	Lattice structure unchanged	None
Faceting and plain scaling	Lattice is divided	None
Pruning and restricting	Some concepts are removed	Possible
Decomposition and general scaling	Lattice is divided	Possible

The following four sections provide an overview of the four types of data weeding techniques. Section 6 discusses the applicability of these techniques to a lexical database of Roget's Thesaurus.

2 Visual Reduction Techniques

The first set of data weeding techniques are visual reduction techniques, which change how the data is displayed without modifying the underlying structure. Many FCA techniques for visual reduction have been described in the literature:

Clarifying and reducing (Ganter & Wille, 1999) result in omitting labels (and the corresponding objects and attributes) without changing the lattice structure. A clarified lattice has at most one object and at most one attribute attached to each concept in the lattice diagram. In the example at the top of Fig. 1, the concept furthest to the left has two objects ("daffodil" and "sunflower") and the concept furthest to the right has two attributes ("purple") and ("dark red"). In the clarified version, "sunflower" and "purple" are removed. If the clarification is achieved using FCA software, then usually the object/attribute that is listed first in the formal context is retained. This choice may not be optimal with respect to the application.

Reducing will further remove all objects that are not at join-irreducible concepts and all attributes that are not at meet-irreducible concepts. This means that in the lattice diagram objects are only attached to nodes that have exactly one edge coming from below and attributes are only at nodes that have exactly one edge from above (Fig. 1). Again, reduction may not be meaningful in all applications. In some applications where the data consists of exemplars and features, the remaining attributes, after reduction, can be considered characteristic, defining features and the remaining objects prototypical examples of the data because of their location at meet- or join-irreducible concepts.

The display of labels and nodes in the diagram can be modified by using **lists, counts and colours**. While it is possible in a manually drawn lattice diagram to carefully place each label in a position where it does not overlap with anything else, the automatic placement of labels in a non-overlapping manner is a major challenge for FCA software. Thus, visual reduction strategies for the display of labels are beneficial, both for

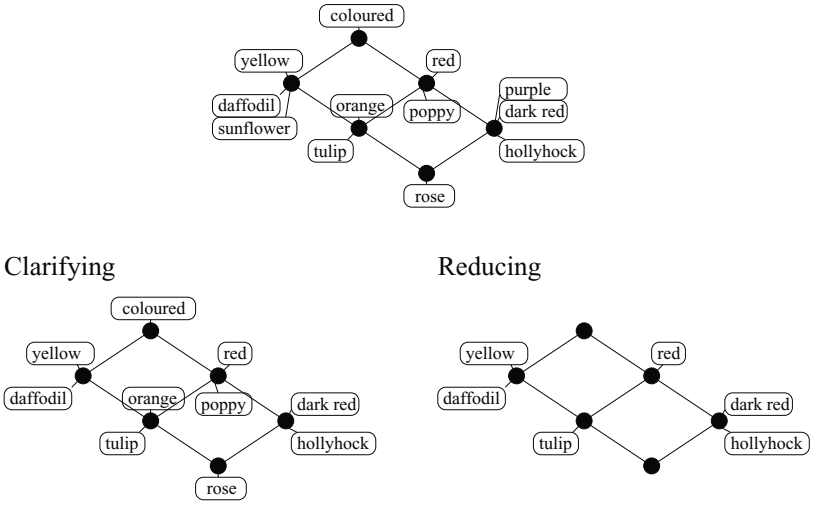


Fig. 1. Visual reduction techniques: clarifying and reducing

the algorithms used by the software and for the users who will be provided with diagrams that are easier to read. Different FCA software tools use different strategies for visual reduction. The FcaStone[2] software, for example, provides an option to draw the concept nodes as boxes which contain lists of objects and attributes truncated to 30 characters. In ToscanaJ[3], objects and attributes can be displayed as a count (showing the number of objects and attributes belonging to a concept instead of their names) or as a scrollable list. These methods reduce the physical space that is occupied by the labels in the display. There are different methods for how the objects and attributes of each concept are counted: either as absolute counts or as relative frequencies; either containing the full extents and intents; or only counting the objects/attributes that are directly attached to a concept in the diagram. The nodes in ToscanaJ can be coloured based on a count of the objects. In the ConExp software[4], a similar effect is achieved by varying the size of the nodes instead of using colours as in ToscanaJ. ConExp uses colours also to show whether a node has objects or attributes directly attached. This feature is especially useful if the display of the labels is turned off.

Some FCA applications do not use lattice diagrams at all, but instead use the lattice structure only for internal algorithms or use **textual displays**. An example is the Credo[5] software. Credo is a meta search engine that calculates a concept lattice for the results of a Yahoo query. The lattice is not displayed graphically but instead as an expandable tree hierarchy. Any lattice can be displayed as a tree by creating multiple copies of any node that has more than one upper neighbour as shown in Fig 2. The first display in Credo consists of the top node of the lattice and its immediate lower neighbours, which

[2] http://fcastone.sourceforge.net
[3] http://toscanaj.sourceforge.net
[4] http://conexp.sourceforge.net
[5] http://credo.fub.it

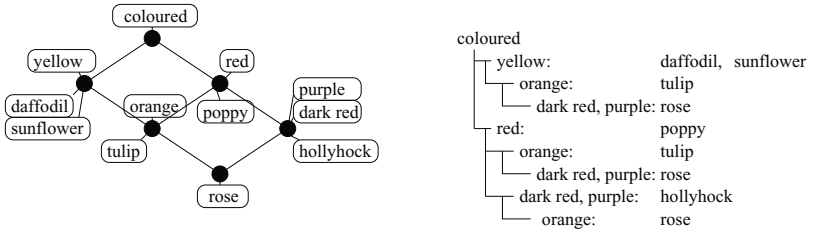


Fig. 2. Representing a lattice as a tree hierarchy

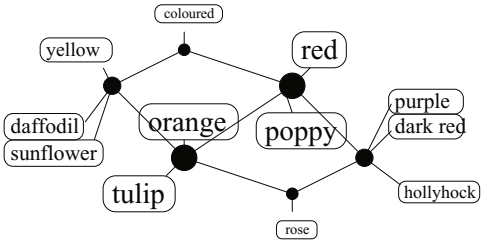


Fig. 3. Fish-eye visualisation

are displayed as a bulleted list that is slightly indented compared to the top node. If a user clicks on any of the nodes, the immediate neighbours underneath this node are displayed as a further indented list (but only up to three levels deep). Clicking on the top node will collapse the expanded sublist. Credo's display is similar to tree displays used for directory listings of folders and files on a computer. Most users will be familiar with such displays. The drawback for these approaches is that the full lattice structure is not immediately visible. Only the relationship between a node and its immediate neighbours is presented.

Fish-eye and zoom are graph visualisation techniques that are used in many graphical displays. Fish-eye displays (Furnas, 1981) enlarge a focal point of a display while gradually reducing the rest of the display (as in Fig. 3). The idea is that users can move the enlarged focal area across a display similar to using a magnifying glass. The focal area can be read in detail while the non-focal area provides structure without detail. The use of fish-eye views for concept lattices was first suggested by Godin et al. (1989) and implemented by Carpineto & Romano (1995). The notion of **zooming** is used with different meanings with respect to concept lattices. In FCA software, "zooming" usually means to enlarge the display. If the lattice is larger than the display window, only parts of the lattice will be visible after zooming in. Thus, this is similar to fish-eye displays except that the non-focal parts are omitted instead of reduced. The notion of "zooming" is also used in a different meaning for the navigation between scales as discussed in the next section.

Moving displays are the final visual reduction technique mentioned in this section. Moving the nodes in a lattice diagram can significantly change the visual complexity of the diagram. From some "angles" a lattice may have a much more complicated display (i.e., more edge crossings, less symmetry) than from other angles. Many FCA tools let

users move individual nodes of the lattice. Some FCA software allows for lattices to be rotated. Other FCA software lets users move parts of a lattice (ideals or filters) at the same time, which helps in the detection of symmetries and other structures in the lattice. (More details and comparisons of FCA software can be found in Tilley (2004) and Priss (2008a)).

3 Faceting and Plain Scaling

Apart from visual reduction techniques that modify the display of the lattices, but not their internal structures, there are data weeding techniques that do change the lattice structures. **Faceting and plain scaling** are techniques which subdivide the set of attributes into smaller sets. The lattice of such a smaller set is traditionally called a "scale" (Ganter & Wille, 1999). Different scales can be combined in nested lattice diagrams and can be interactively explored in Toscana systems (Vogt & Wille, 1995). In this section, only scaling of "single-valued contexts" is considered. Priss (2008b) argues that the idea of "plain scaling" as described in FCA occurs in several other disciplines under other names. In particular, plain scaling is very similar to the idea of "faceting" in library and information science. For both scales and facets it is usually required that the subdivisions are carefully selected (usually manually) and meaningful. Meaningful subdivisions will group attributes based on their types, shared features, and so on. For example, one scale could contain size attributes while another contains colour attributes. Each scale then arranges the objects based on its particular aspect or viewpoint.

As mentioned above, the notion of "zooming" is used to describe the navigation from an "outer scale" into an "inner scale" in a nested lattice diagram in Toscana systems (Vogt & Wille, 1995). Roth et al. (2008) modify this notion of "zoom" by applying local criteria to the calculation of inner scales instead of a global algorithm. In Stumme's (1996) approach to "local scaling" some concepts are expanded with inner scales, while others are not.

Historically, faceted systems in library science have been less popular with end users because it can be difficult to use such systems in a paper-based environment. To some degree facets or scales are only really useful if they are presented via a software interface which lets users explore the relationships interactively. This is because it is impossible to display all of the facets simultaneously for all but the smallest data sets. As a rule of thumb, three levels of nesting (or three scales) is the maximum that can be shown simultaneously. Thus, the information contained in a formal context and its scales is best explored interactively by starting with a few scales, deciding how to nest them, and then navigating from one scale to the other.

Besides the Toscana systems, as described by Vogt & Wille (1995), there have been recent software developments in the library and information science community which are potentially interesting for FCA because of the similarity between scales and facets. This has been discussed in detail by Priss (2008b). Such software often presents different scales side by side instead of nested. The effect of user selections on one scale is instantly applied to the other scales by the software. It would be interesting to conduct a user study comparing different techniques of representing scales. This could lead to new developments in FCA software.

4 Pruning and Restriction

Data weeding techniques of **pruning** reduce a lattice in a manner that is usually based on statistical measures. The effect is a removal of concepts usually from the bottom of the lattice. In many cases the lattice structure is changed significantly. Some forms of pruning result in ordered sets which are not lattices.

Kuznetsov (2007) defines a notion of "stability" of a formal concept based on his earlier work on stability in similarity operations (Kuznetsov, 1990). Roughly, the stability index of a concept C is based on counting the number of those subsets of the extent of C whose intent equals the intent of C, divided by some number related to the size of the extent. The idea behind this is to determine how many objects in the extent are necessary and sufficient to creating the concept. This is because a concept can also have many objects that are shared with many other concepts and are not defining for this particular concept. The ConExp software (mentioned above) implements a similar but slightly different notion of "stability" which, according to ConExp's user guide, calculates for each concept the minimal number of objects needed to be removed so that the intent of this concept disappears from the concept lattice.

Pruning is then the process of removal of "less stable" concepts based on any of the stability notions. All concepts that are less stable than a user-defined threshold are removed. Roth et al. (2008) use a slight variant of Kuznetsov's definition of stability for their pruning, in combination with their notions of nesting and zooming. As mentioned above, Roth et al.'s notion of "zooming" is slightly different from Vogt & Wille's (1995). They divide the set of attributes based on preferences and use attributes that are considered more important in outer scales. The inner scales are then "pruned" individually, using local instead of global criteria. Belohlavek & Sklenar (2005) propose a different method of pruning based on attribute dependency formulas, which uses an expert-specified hierarchy on the set of attributes.

Last but not least, an iceberg lattice (Stumme et al., 2002) is an order filter consisting of the top most concepts of a concept lattice. Only those concepts are included whose concepts have a "support" that is higher than a threshold. These concepts are called "frequent". The support is calculated as the number of objects of the concept divided by the total number of objects in the formal context. The notion of iceberg lattices was developed in the framework of data mining and is based on what is called "frequent itemsets" in data mining terminology. A variation of this approach is to also include the immediate lower neighbours of the frequent concepts, which may result in an ordered set that is not an order filter.

Old (2003) describes the notion of **restricting** concept lattices, with respect to neighbourhood lattices. Neighbourhood lattices were first described by Rudolf Wille in an unpublished manuscript and first published by Sedelow & Sedelow (1993) in the context of lexical databases. The operation which underlies the selection of elements in a neighbourhood has been called the "plus operator" (Priss & Old, 2004) as opposed to the "prime operator" used in concept formation. This is because the prime operator applied to a set of objects selects all attributes which are shared among *all* objects in the set, whereas the plus operator selects all attributes that belong to *at least one* of the objects in the set. For a large formal context A, the plus operator can be used to derive a smaller "neighbourhood context" B as follows: a "n-m-neighbourhood" starts with

an object from A and has the plus operator applied $(2n-2)$-times to obtain the set of objects of B and $(2m-1)$-times to obtain the set of attributes of B. This context B represents the neighbourhood of the object that was used at the start of the operation. Because the plus operator is not a closure operator, a few iterations of the operator can result in a context B whose size is fast approaching the size of A. Therefore Old (2003) experiments with "restricted" neighbourhood lattices which modify the plus operator by selecting objects (attributes) which have at least two (three, etc) attributes (objects) instead of at least one.

5 Decomposition and General Scaling

Ganter & Wille (1999) describe numerous methods for deriving parts and **decompositions** of concept lattices. The methods that are used in actual applications and that are implemented in FCA software are usually only the simplest examples of such constructions. For example, a horizontal decomposition of a lattice is a decomposition into components whose horizontal sum (Ganter & Wille, 1999) is the original lattice. A horizontal decomposition of a lattice refers to the components that a lattice falls into after removing the top and bottom concept. If a plus operator is applied until the sets do not change any further, it yields a horizontal decomposition of the original lattice (Priss & Old, 2006). Horizontal decompositions have been used in software analysis (Snelting, 2005) and other applications. A form of decomposition is also the Semantic Mirrors method. This method was invented by Dyvik (2004) and translated into FCA terminology by Priss & Old (2005). The Semantic Mirrors method is similar to a form of repeated applications of decomposition.

In contrast to plain scaling, in **general scaling** the scales are combined using a "composition operator". This appears to be mostly of theoretical interest. We are not aware of any software implementations of scaling techniques other than plain scaling. Plain scaling is often applied to many-valued contexts, which are transformed into single-valued contexts via the scales. This does imply loss of information.

6 Data Weeding Techniques for Roget's Thesaurus

Before discussing which of the data weeding techniques described above are relevant for Roget's Thesaurus, it shall be explained why Roget's Thesaurus is of interest. Roget's Thesaurus (1911, 1962) is a semantic dictionary that is organised by concepts, rather than words, into a classification tree. The explicit structure of the book consists of three main parts: the top level of the hierarchy represented by the tabular Synopsis of Categories; the Sense Index which continues the hierarchy down to the lowest level; and the Word Index which lists the words in alphabetic order along with their senses ordered by part-of-speech. The senses are represented in the Word Index as references to locations in the Sense Index.

The Sense Index lists the 1,000 or so categories representing the notions found at the most detailed level of the Synopsis. Categories generally occur in pairs as opposed notions, or antonyms. Each category is subdivided into paragraphs which contain groups of words at the lowest level. The words in each group at the lowest level are considered

"synonyms" with respect to the thesaurus structure. Each group of synonyms denotes a "sense". A particular occurrence of a word is also called an "entry" of the thesaurus. The notation for senses used in this paper (e.g. 227:1:1) consists of the category number (227), followed by the paragraph number (1) and the synonym group number (1).

Roget's Thesaurus has been studied or used for the automatic classification of text, automatic indexing, natural language processing, word sense disambiguation, semantic classification, computer-based reasoning, content analysis, discourse analysis, automatic translation, and a range of other applications by many different researchers (cf. Old (2003 and 2004) and Priss & Old (2009) for more details). The reason why the Thesaurus has been used in such applications is that it contains an implicit conceptual structure based on the polysemy and synonymy relationships between the word entries and their senses. However, although the explicit structure of Roget's Thesaurus is evident to any reader, the implicit, hidden, or "inner structure" (Sedelow, 1988) is not. FCA can be a useful tool in exploring this inner structure because the relationship between words and senses for the Thesaurus is a very large formal context. Each "entry" corresponds to a cross in this context. Since the formal context of Roget's Thesaurus has about 113,000 objects, 71,000 attributes and 200,000 crosses, some data weeding technique is required in order to extract smaller-sized lattices.

Neighbourhood lattices, as described in the previous section, have been used for the exploration of Roget's Thesaurus since Sedelow & Sedelow (1993). Fig. 4 shows a 2-1 neighbourhood for the word "think" (i.e., the plus operator has been applied twice to retrieve the objects and once to retrieve the attributes). Neighbourhood lattices can be of very different sizes. We have calculated the sizes of all 2-1 and 2-2 neighbourhoods of common words in Roget's Thesaurus (using Roget (1962)). The neighbourhoods range from single-concept lattices (for words such as "amoeba") to a lattice with 118 concepts (713 concepts) for the word "cut" in the case of 2-1 neighbourhoods (2-2 neighbourhoods, respectively). Lattices with less than 5 concepts are not very interesting whereas lattices with more than 35-40 concepts tend to be too large to be visualised. Therefore, different words require different types of neighbourhoods in Roget's Thesaurus. About 100 words (mostly verbs of Anglo-Saxon origin) have 2-1 neighbourhood lattices with more than 35 concepts. These could be benefit from using data weeding techniques.

The on-line interface at `www.roget.org` lets a user explore Roget's Thesaurus by entering a word and then viewing a neighbourhood lattice of that word. The lattices are generated in run-time. The remainder of this section investigates which data weeding techniques are appropriate for automatically adjusting the size of the lattices. Visual reduction techniques depend on the display software that is used because they do not affect the structure of the lattices. The on-line interface uses the "concept as boxes" display feature of FcaStone[6] in order to avoid overlapping labels. Scaling and faceting often depend on manual subdivision of the set of attributes, which is not applicable in this case. Thus, the techniques that are of interest are pruning, restriction and decomposition.

Figs. 5 and 6 show a pruned lattice and an iceberg lattice, respectively, for the example from Fig. 4. Unfortunately, pruning and iceberg lattices appear not to be appropriate data weeding techniques for Roget's Thesaurus. This is because neighbourhood lattices are always of a particular structure. Neighbourhood lattices of type 2-1

[6] `http://fcastone.sourceforge.net`

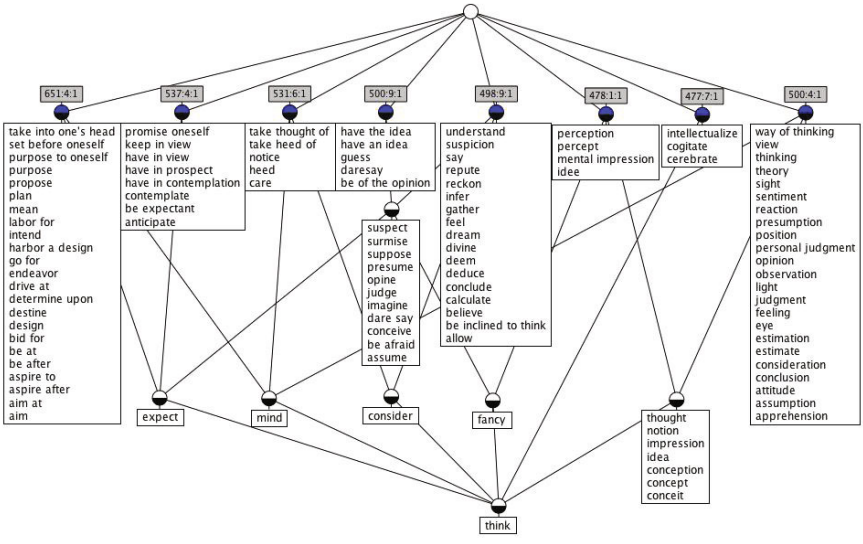


Fig. 4. A neighbourhood lattice from Roget's Thesaurus for the word "think"

always have the original word attached to the bottom node. They normally contain many more objects than attributes (because the plus operator has been applied twice to retrieve the objects). Most of the objects that were added in the last step in a 2-1 neighbourhood lattice are attached to the nodes adjacent to the top node because their distinguishing attributes are not yet part of the neighbourhood context. Dually, most of the attributes in a 2-2 neighbourhood lattice will be attached to nodes that are adjacent to the bottom node. Clearly, the most interesting objects in Fig. 4 are "expect", "mind", "consider", "fancy", and "thought, notion, ..." because they share more than one sense with "think". But these objects are exactly the ones that are pruned away in Figs. 5 and 6. Because of the particular structure of neighbourhood lattices, pruning and iceberg lattices are not appropriate. Pruning and iceberg lattices are based on the relative sizes of the extents, but for neighbourhood lattices these sizes are distorted by the algorithm.

In a similar fashion, the Semantic Mirrors method (see above) is not of interest for these kinds of lattices because this method essentially looks for symmetries between the objects and attributes. This works well for neighbourhood lattices that are derived from bilingual data (the objects are words from one language, while the attributes are words from a second language). But it does not yield interesting results for Roget's Thesaurus because the structures among words are quite different from the structures among senses. The Semantic Mirrors method applied to the lattice in Fig. 4 would result in a decomposition of the lattice into single-concept lattices. On the other hand, other forms of decomposition might be of interest. A horizontal decomposition separates the concept with the sense 477:7:1 in Fig. 4 from the rest of the lattice. If horizontal decomposition results in more than one component, this can (but does not have to) indicate that a word

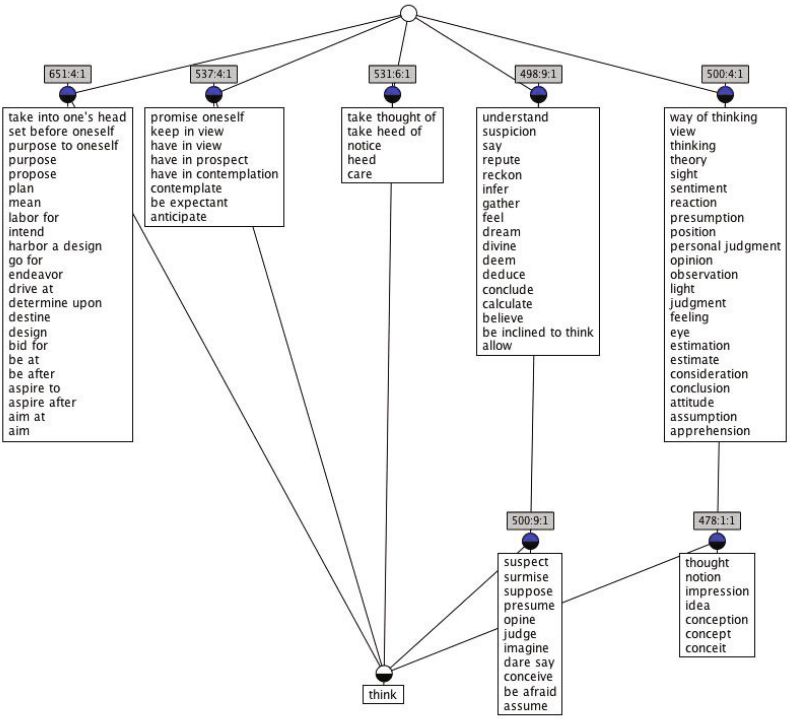


Fig. 5. A pruned lattice showing he most stable concepts (using ConExp) of Fig. 4

is a homograph (Old, 2006), i.e. that the word has two completely unrelated senses. In this case, sense 477:7:1 is a polysemous sense of "think", but not a homograph.

The most promising data weeding technique for neighbourhood lattices of Roget's Thesaurus appears to be restriction (Old, 2003). Fig. 7 shows the restricted lattice of Fig. 4. This lattice maintains the main structures from Fig. 4, but all objects that do not share at least two senses with "think" have been removed. Different degrees of restriction are possible: only keeping words (or senses) that have at least 3 senses (or words), and so on. Incidentally, in retrospect Priss & Old (2009) discovered that the restriction algorithm is very similar to techniques developed for Roget's Thesaurus by the Cambridge Language Research Unit in the 1950s.

The goal for the on-line interface at `www.roget.org` is to generate neighbourhood lattices that are of appropriate sizes and calculated sufficiently fast. A simple strategy could be to use the number of crosses in the context as a rule of thumb for predicting the size of the neighbourhood lattices (for example, if the context has more than 100 crosses, then use restriction). Another simple strategy would be to apply restriction if the smaller of the sets of objects and attributes is larger than 20 and the larger one is larger than 80. These strategies have been tried and found to yield reasonably good results for Roget's Thesaurus.

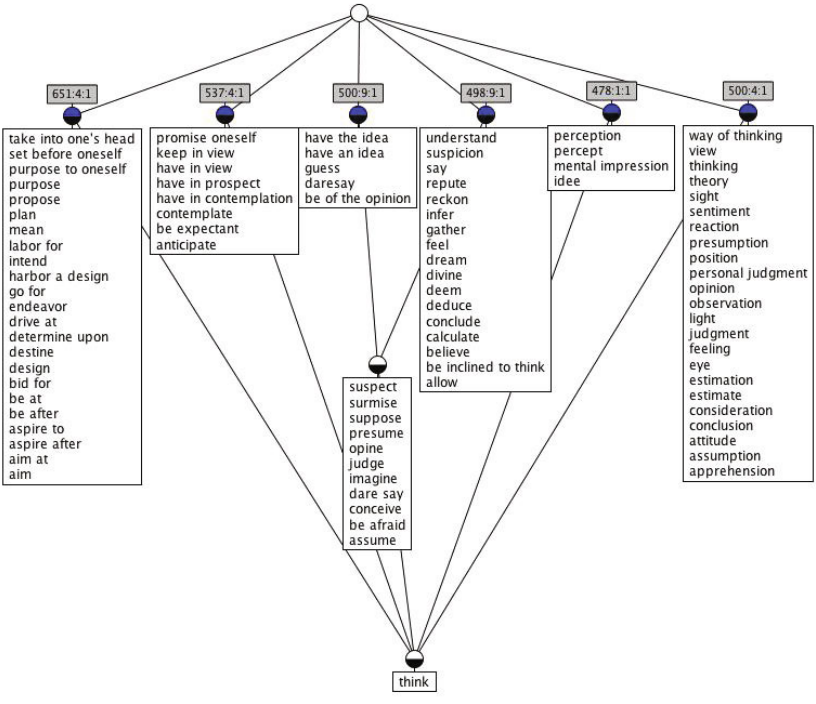


Fig. 6. An iceberg lattice for the lattice in Fig. 4

A third, more precise strategy is to calculate the sizes of all neighbourhood lattices (2-1, 2-2 neighbourhoods and restriction) in an off-line mode and to store the results in a database table. Calculating the number of concepts can be slow for large lattices, but because they need to be calculated only once that is not a problem. When a user enters a word into the interface, the size of its neighbourhood lattice can be looked up and be used to select a type of neighbourhood with an appropriate size. Only about 6000 words in Roget (1962) have 2-2 neighbourhood lattices with more than 20 concepts. For about 500 of these words, the 2-1 neighbourhood lattice also has more than 20 concepts. When applying restriction, only about 60 words are left which have more than 30 concepts in their neighbourhood lattices. For the word “cut” which has the largest concept neighbourhood of all words in the Thesaurus, a 2-1 neighbourhood needs to be restricted to objects and attributes that have at least 3 crosses in the context in order to reduce the size of the lattice to 38 concepts. This is a size that can still be graphically represented. At the other extreme, considering lattices that might be too small, according to Priss & Old (2006) there are about 20,000 words for which the lattice never has more than one concept for any type of neighbourhood. But these words include archaic, foreign and specialist words and phrases, which are unlikely to be entered by users in the on-line interface.

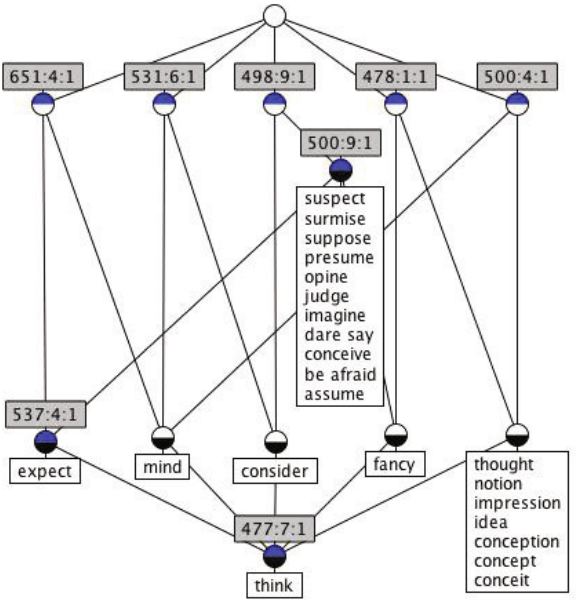


Fig. 7. A restricted neighbourhood lattice

7 Conclusion

In summary, data weeding techniques select data from a given set based on the specific needs and purpose of an application. Visual reduction techniques do not change the lattice structure and are mostly a feature of the software interface that is used to draw and display the lattices. Scales and facets rely on meaningful subdivisions of the set of attributes that are often manually derived and not ideally suited for automatically generated lattices. Although pruning is very useful in other applications (for example for clustering), it appears to prune away the wrong concepts with respect to neighbourhood lattices. Some forms of decompositions might be applicable to neighbourhood lattices, but this needs to be investigated further. Experimentation has shown that restriction is useful for neighbourhood lattices because it maintains most of the structure of the lattices while removing objects and attributes.

In general, data weeding techniques appear to be very much dependent on the type of application. There may not be a single type of technique that is suitable for all kinds of lattices. An interesting topic for further research would be to determine whether it might be possible to develop some guidelines that predict which data weeding techniques are appropriate for which types of lattices.

References

1. Belohlavek, R., Sklenar, V.: Formal Concept Analysis Constrained by Attribute-Dependency Formulas. In: Ganter, B., Godin, R. (eds.) ICFCA 2005. LNCS (LNAI), vol. 3403, pp. 176–191. Springer, Heidelberg (2005)

2. Carpineto, C., Romano, G.: Ulysses: a lattice-based multiple interaction strategy retrieval interface. In: Blumenthal, B., Gornostaev, J., Unger, C. (eds.) EWHCI 1995. LNCS, vol. 1015, pp. 91–104. Springer, Heidelberg (1995)
3. Dyvik, H.: Translations as semantic mirrors: from parallel corpus to wordnet. Language and Computers 49(1), 311–326 (2004)
4. Furnas, G.W.: The FISHEYE View: A New Look at Structured Files. Bell Laboratories Technical Memorandum (1981)
5. Ganter, B., Wille, R.: Formal Concept Analysis. Mathematical Foundations. Springer, Heidelberg (1999)
6. Godin, R., Gecsei, J., Pichet, C.: Design of browsing interface for information retrieval. In: Belkin, N.J., van Rijsbergen, C.J. (eds.) Proc. SIGIR 1989, pp. 32–39 (1989)
7. Kuznetsov, S.O.: Stability as an estimate of hypotheses based on similarity operation (in Russian). Nauchno-Tekhnicheskaya Informatsiya (NTI) 2, N12, 21–29 (1990)
8. Kuznetsov, S.O.: On Stability of a Formal Concept. Annals of Mathematics and Artificial Intelligence 49, 101–115 (2007)
9. Old, L.J.: The Semantic Structure of Roget's. A Whole-Language Thesaurus. PhD Dissertation. Indiana University (2003)
10. Old, L.J.: Unlocking the Semantics of Roget's Thesaurus. In: Eklund, P. (ed.) ICFCA 2004. LNCS (LNAI), vol. 2961, pp. 244–251. Springer, Heidelberg (2004)
11. Old, L.J.: Homograph Disambiguation using Formal Concept Analysis. In: Missaoui, R., Schmidt, J. (eds.) Formal Concept Analysis. LNCS (LNAI), vol. 3874, pp. 221–232. Springer, Heidelberg (2006)
12. Priss, U., Old, L.J.: Modelling Lexical Databases with Formal Concept Analysis. Journal of Universal Computer Science 10(8), 967–984 (2004)
13. Priss, U., Old, L.J.: Conceptual Exploration of Semantic Mirrors. In: Ganter, B., Godin, R. (eds.) ICFCA 2005. LNCS (LNAI), vol. 3403, pp. 21–32. Springer, Heidelberg (2005)
14. Priss, U., Old, L.J.: An application of relation algebra to lexical databases. In: Schaerfe, H., Hitzler, P., Ohrstrom, P. (eds.) ICCS 2006. LNCS (LNAI), vol. 4068, pp. 388–400. Springer, Heidelberg (2006)
15. Priss, U.: FCA Software Interoperability. In: Belohlavek, K. (ed.) Proceedings of the Sixth International Conference on Concept Lattices and Their Applications (CLA 2008), pp. 133–144 (2008a)
16. Priss, U.: Facet-like Structures in Computer Science. Axiomathes 14, 243–255 (2008b)
17. Priss, U., Old, L.J.: Revisiting the Potentialities of a Mechanical Thesaurus. In: Ferré, S., Rudolph, S. (eds.) ICFCA 2009. LNCS, vol. 5548, pp. 284–298. Springer, Heidelberg (2009)
18. Roget, P.M.: Roget's International Thesaurus, 3rd edn. Thomas Crowell, New York (1962)
19. Roget, P.M.: Roget's Thesaurus (1911), Available from the Project Gutenberg `http://promo.net/pg`
20. Roth, C., Obiedkov, S., Kourie, D.G.: On succinct representation of knowledge community taxonomies with formal concept analysis. International Journal of Foundations of Computer Science (IJFCS) 19(2), 383–404 (2008)
21. Sedelow Jr., W.A.: Computer-based planning technology: an overview of inner structure analysis. In: Sixth Annual Conference on New Technology and Higher Education: Acquisition, Integration, and Utilization (1988)
22. Sedelow, S., Sedelow, W.: The Concept "concept". In: Proceedings of the Fifth International Conference on Computing and Information, Sudbury, Ontario, Canada, pp. 339–343 (1993)
23. Snelting, G.: Concept Lattices in Software Analysis. In: Ganter, B., Stumme, G., Wille, R. (eds.) FCA 2005. LNCS (LNAI), vol. 3626, pp. 272–287. Springer, Heidelberg (2005)

24. Stumme, G.: Local Scaling in Conceptual Data Systems. In: Eklund, P., Mann, G.A., Ellis, G. (eds.) ICCS 1996. LNCS, vol. 1115, pp. 308–320. Springer, Heidelberg (1996)
25. Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., Lakhal, L.: Computing Iceberg Concept Lattices with Titanic. J. on Knowledge and Data Engineering 42(2), 189–222 (2002)
26. Tilley, T.: Tool Support for FCA. In: Eklund, P. (ed.) ICFCA 2004. LNCS (LNAI), vol. 2961, pp. 104–111. Springer, Heidelberg (2004)
27. Vogt, F., Wille, R.: TOSCANA - a graphical tool for analyzing and exploring data. In: Tamassia, R., Tollis, I.G. (eds.) GD 1994. LNCS, vol. 894, pp. 226–233. Springer, Heidelberg (1995)

Virtual Catalog: The Ontology-Based Technology for Information Retrieval*

Dmitry E. Palchunov

Institute of Mathematics SB RAS,
Novosibirsk State University
Novosibirsk, Russia
palch@math.nsc.ru

Abstract. This paper is devoted to information retrieval and fine search organization on the Internet. The presented approach is based on model-theoretical formalization of a subject domain ontology.

The formal measures of information retrieval effectiveness are studied. The central problem of our consideration is the problem of pertinence of information retrieval. To contribute to the solution of this problem, we present an approach which is called the Virtual Catalog. This approach is a synthesis of search engines and Internet catalogs.

The development of the Virtual Catalog is based on subject domain ontologies. Ontologies are formalized in model-theoretical terms. We develop methods of ontology representation as a network of sentences of first-order predicate logic. We use three types of ontology to solve the problem of pertinence of information retrieval: ontology of the subject domain which is relevant to the desired information; ontology of the Internet which describes various types of Internet resources; and ontology of user needs and types of information search tasks. With the help of the subject domain ontology, we specify the user's field of interest. The user needs ontology helps us to specify what kind of information the user wants. The Internet ontology gives the possibility to determine the desired type of Internet resource. Our approach is applied to develop Virtual Catalogs for two fields: mathematics and information security. We present technologies for constructing ontologies for these fields and technologies for fine search organization on the Internet.

1 Introduction

It is obvious, that a huge volume of information is presented on the Internet. This information concerns all kinds of human activity — all fields of science and technology, all branches of industry and business, rest, sport, culture, art, etc. It is possible to say with confidence that, on an overwhelming number of questions, the answers which are already known by mankind can be found in documents presented on the Internet.

* Supported by RFBR grant N 05-01-04003-NNIO-a (DFG project COMO, GZ: 436 RUS 113/829/0-1) and by the Federal Agency for Science and Innovations of the Russian Federation, the State contract N P-1008.

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 164–183, 2011.

Nevertheless, till now, the Internet may be more likely to be named a hiding place of treasures than a universal problem solver. The problem is that, though a full and exhaustive answer to a given question is contained in the Internet, it is not clear how we can extract this answer. Especially, it concerns the specific information which is presented by a small number of Internet resources, for example, scientific and technical information. In spite of the fact that, now, universal search systems as well as specialized search systems have been successfully developed, this problem does not lose acuteness.

In the paper, an approach to the search for scientific and technical information, based on the development of metasearch systems of a special kind — Virtual Catalog — is presented. Now, Virtual Catalogs on mathematics and information security are developed. The purpose of the Virtual Catalog is to extract from the Internet any information concerning the given subject domain, which is necessary to the user. To achieve efficiency in the information search, we pay special attention to the set of tools which our search system gives to the user, such that he could formulate his information needs precisely and completely. The elaboration of the toolkit for the formalization of the search query is the most important point of our research.

The development of the Virtual Catalog is based on the model-theoretical approach to the formalization of ontologies. In the paper, the basic concepts and results of model-theoretical formalization of ontologies are stated. In particular, the question of the possibility of representation of ontological information by various kinds of glossary is studied.

Methods of automation of the construction of ontologies, based on the extraction of ontological knowledge from natural language texts, are developed for the realization of Virtual Catalogs.

2 Information Retrieval on the Internet

2.1 Measures of Efficiency of Information Retrieval

To determine the quality of information retrieval, to compare the results of the work of various search engines and to carry out of the testing of the search system developed by us, we need formal measures of quality of information retrieval.

As the formal description of information retrieval, we start with three components of this process. The first component is the user who intends to obtain some information. The second component is the formalized query for the search engine. The third component is the answer to this query: the list of results of the search engine.

By ***pertinence*** of information retrieval, we mean a function with two arguments: the first argument of this function is the information need of the user, which has been formalized in the query; the second argument is the ordered list of Internet resources which the search engine has given out in reply to this query. For simplicity and convenience, we suppose that this function possesses values in the interval [0, 1]. If the list of results is more relevant to

the information need of the user, then the value of the function is greater. Full success is 1; absence of any success is 0.

Note that, by our definition of pertinence (unlike another, also widely used, definition of this notion), the value of pertinence does not directly depend on how interesting the obtained information is to the user. For example, if the user tries to find one kind of information (and makes a formal query), and the engine gives him absolutely another kind of information then, by our definition, pertinence will be equal to 0 irrespective of whether the other information is interesting to the user or not.

Pertinence of information retrieval depends on two conditions: how precisely the formal query, created by the user, agrees with the information need which he wished to formalize, as well as how well the results of the search engine agree with the formal query.

Thus, pertinence may be decomposed into the sum (or, in our case — into the product) of two components, which we call *adequacy* and *relevance*.

By ***adequacy*** of information retrieval, we mean a function with two arguments of which the first argument is the information need of the user, which has been formalized in the query, and the second argument is the formal query. The function possesses values in the interval [0, 1]. It represents how well the formal query meets the information need which the user tried to formulate: full coincidence is 1; absence of coincidence is 0.

By ***relevance*** of information retrieval, we mean a function with two arguments: the first argument of this function is the formal query and the second argument is the ordered list of Internet resources which the search engine gives out in reply to this query. It shows how well the list of results conforms to the formal query. The function possesses values in the interval [0, 1].

For simplicity, we suppose that pertinence is equal to adequacy multiplied by relevance. Or, more precisely, adequacy is equal to pertinence divided by relevance — from our point of view, pertinence and relevance are more primary parameters than adequacy.

At the present time, the most developed measure of information retrieval is relevance. Modern search engines attain a high value of this parameter.

The numerical score of relevance depends on three measures — precision, recall and ranking. ***Ranking*** is the correctness of the order in which the list of results of information retrieval is presented. ***Precision*** is the fraction of a search output that is relevant to a formal query. So, precision is equal to the number of documents which are relevant to a formal query, retrieved by a search engine, divided by the total number of documents retrieved. ***Recall*** is the ability of a search engine to obtain all or most of the relevant documents presented on the Internet. Recall is equal to the number of relevant documents retrieved by a search engine, divided by the total number of existing relevant documents on the Internet, which should have been retrieved.

The main aim of the methods of information retrieval developed by us is high pertinence. To achieve high pertinence, we should achieve both high adequacy and high relevance.

In turn, to attain high adequacy, we should give to the user a good set of tools for creating a formal query. These tools should possess the following important qualities:

1. Great expressive force: the ability to represent various information needs.
2. Clarity for the user. The sense of the created formal query should be completely clear to the user and should not allow ambiguities.
3. The creation of the formal query should not take too much time.
4. The use of tools for the creation of the formal query should not demand special training or education.

The specified requirements, at first sight, are very strong and, to some extent, inconsistent. However, their full or even sufficient satisfaction will give to ordinary users of the Internet really flexible and effective tools of information retrieval.

Consider these requirements in more detail. For most modern search engines, such as Google, AltaVista, Yahoo, Lycos, AllTheWeb, etc., the usual search tool is a combination of words. Some search engines use advanced features such as Boolean searching or date ranges. Such tools for creating a formal query, obviously, do not satisfy requirement 1, but do satisfy requirement 2 and, at first sight, requirements 3 and 4. However, the search for specific and rare information by means of the simple input of a sequence of a few words requires a lot of time: the user should test many variants of queries. Therefore, condition 3 is not satisfied. Moreover, a successful result of a search for rare information requires the user to have experience and search skills. So, condition 4 is not satisfied either.

Strictly speaking, for such simple kinds of formal query, the measure of pertinence has no sense, because the same sequence (containing three to five words) may be inputted by users having absolutely different information needs. Therefore, only relevance is sensible for almost all search engines. As it was mentioned above, most search engines successfully achieve good relevance.

Other types of information systems, which give users links to various electronic documents, are Internet catalogs, universal or specialized. The interface of an Internet catalog certainly meets requirements 2–4. An evident defect of catalogs is the small amount of resources presented by them, compared with the total number of resources on the Internet, and the absence of the newest, fresh information.

As regards item 1 of the requirements — the ability to represent various information needs — catalogs solve this problem in a rather one-sided manner. Each catalog represents just one kind of search task. For example, "to find software with certain properties" (www.download.com, www.tocows.com), or "to find a simple explanation of a notion which is unknown to the user" (www.wikipedia.org), etc. Thus, Internet catalogs enable quick and exact specification of the search problem, but from a very narrow spectrum. Internet catalogs attain high pertinence — each catalog for a narrow class of search tasks.

2.2 The Search for Scientific and Technical Information on the Internet

For scientific investigations, it is necessary to improve the possibilities of obtaining new scientific and technical information. This problem is most critical for those areas where there is a permanent inflow of new information, and thus information quickly becomes outdated: for example, in the field of information security. Now, the most pervasive source of new scientific and technical information is the Internet.

For today, there are various specialized search systems representing scientific and technical Internet resources. For example, Scirus has more than 250 million indexed pages. This system is universal and has a very important feature: in "Advanced search" it enables the user to specify the kind of resource. Besides this, it is possible to narrow the subject domain for the desired information. Scirus is very developed and, in a certain sense, is a prototype of the Virtual Catalog developed by us.

There are many other scientific search systems and catalogs: universal — Google Scholar, SciNet, ScienceDirect, Science Research Portal, Windows Live Academic, CiteSeer, InfoTrieve, Scientopica, HighWire Press, etc., and specialized — Zentralblatt MATH, MathSearch, EULER, ChemIndustry, Medline and so forth.

The main weakness of practically all such search systems is that the search query is a small sequence of keywords. By means of these keywords, it is necessary to specify precisely the subject domain in which the search is being carrying out. For an inexperienced user, it is very complicated to specify a narrow subject domain.

Such systems work well in situations when the user knows approximately what document or Internet resource he wishes to find: for example, a known article, book, conference or organization, by the incomplete data which he has. The situation is much worse when the user has another problem: to find new, absolutely unknown information in the given narrow field of science.

For almost all modern search systems, the following takes place:

– In the formulation of the search query, the set of keywords is central.

– There is no precise description which search problems the system can solve effectively.

– Systems solve different search tasks with different degrees of efficiency. Different systems have different sets of such "successful" and "unsuccessful" tasks.

Thus, despite the presence of plenty of specialized search systems, till now, the problem of searching for scientific and technical information is far from the final solution.

3 Mathematical Basis: Model-Theoretical Approach to the Formalization of Ontologies

Our approach to information retrieval on the Internet is based on the use of subject domain ontologies [7–9, 14, 26]. In our search system — Virtual Catalog — we

use ontologies for two aims: first, for tools of formulating the search query, which should possess great expressive force and should be able to express various information needs of the user; second, for the achievement of a relevant reply to the formalized query. Note that, if the search query is more expressive and complicated, then it is more difficult to attain the relevance of the reply to this query.

To solve these problems, we use a hierarchy of ontologies of subject domains, an ontology of types of Internet resources and an ontology of types of search problems.

3.1 Logical Means of Ontology Representation

To automate the development and use of ontologies, it is necessary to have completely formalized means for the representation of ontologies.

Now, the most developed approach to the representation, extraction and automatic processing of knowledge is the project Semantic Web of the WWW consortium [5, 9, 12]. This project is closely connected with knowledge representation based on ontologies. Within the framework of this project, the language of ontology representation OWL — "Web Ontology Language" [13], has been developed. In turn, the language OWL is based on Description Logic [1].

At the moment, as a logical means for the representation of ontologies, the greatest attention of experts is attracted by Description Logic. Description Logic has advantages over other formal means of description and modelling of subject domains. On the one hand, it gives a rich and expressive language for the representation of various information. This language is more expressive than, for example, Prolog. On the other hand, Description Logic is decidable and, moreover, has effective provers — for instance, RACER (Renamed Abox and Concept Expression Reasoner) [10, 11]. Now, Description Logic is the most widespread means of mathematical representation of knowledge for the development of ontologies and their further use.

Within the framework of Semantic Web, the language OWL has been introduced. The development of ontologies in OWL is supported by the program system Protégé. Now, there are a number of other tools for the development and representation of ontologies. These are OilEd, Ontolingua, OntoEdit, WebOnto, OntoSaurus, ODE, etc. All of them are based on various logical means.

What should be the requirements of logical means for the representation of knowledge about subject domains? As the basic requirements, we would mention the following:

– decidability — the existence of algorithms to check the provability and equivalence of formulas and the consistency of finite sets of formulas;
– efficiency of resolving algorithms;
– presence of already developed provers (programs for automatic proving, which carry out logical derivations).

Thus, at the moment, there are many logical means for knowledge representation. However, for the use of these logical means, it does not matter whether the represented knowledge about the subject domain is ontological or not. With

identical success, using these formalisms, it is possible to represent subject domain ontology as well as arbitrary knowledge about the subject domain.

In the elaboration of logical means for the representation of ontologies, in most cases, attention is not paid to the logical analysis of the ontological knowledge about subject domains. Logical methods of extraction of ontological knowledge from natural language texts and their further structurization are poorly developed. However, natural language texts contain a huge amount of ontological information on almost all subject domains. The problem is how this information may be extracted and how to structure the information then. To solve this problem, we develop a model-theoretical approach to the formalization of subject domain ontologies [20–25].

3.2 Model-Theoretical Approach to the Formalization of Ontologies

In this section, we present the model-theoretical approach to the formalization of ontologies, which is the mathematical basis of our investigation.

In the elaboration of the first version of the Virtual Catalog, the most important aspects for us are two kinds of ontological information:

1. Proximity between key concepts.
2. Definition of the sense of key concepts via other key concepts.

Using the structure of proximity between key concepts, we determine how much documents belong to the given subject domain. With the help of the structure of proximity, we automatically generate a query, the reply to which should be relevant to the given subject domain.

Definitions of key concepts are necessary, first, for the explanation of the sense of headings of the Virtual Catalog for the user. Secondly, the presence of exact definitions of key concepts allows us to extract such ontological lexical relations as the synonymy of concepts, the relation "is a kind of", and the relation "concept A is necessary for definition of concept B" and so on. These relations are used in our system for increasing the relevance of the automatically generated answer.

To give a formal definition of the structure of proximity between terms, we introduce the notion of a formal semantic network. We investigate the possibility of representation of arbitrary knowledge about the subject domain by a formal semantic network.

We formalize definitions of key concepts of a subject domain by means of the notion of a formal glossary.

Definitions and denotations on model theory are most important for the subsequent text [4, 6].

By signature we call the tuple $\sigma =< P_1, \ldots, P_n, f_1, \ldots, f_k, c_1, \ldots, c_m >$, where $P_1, \ldots, P_n$ are symbols of predicates, $f_1, \ldots, f_k$ are symbols of functions and $c_1, \ldots, c_m$ are symbols of constants. By $FV(\varphi)$ we denote the set of all free variables of formula φ. $S(\sigma)$ denotes the set of all sentences of the signature σ. The signature of a formula φ, that is, the set of all signature symbols appearing in φ, is denoted by $\sigma(\varphi)$. $\sigma(\Gamma)$ denotes the signature of the set of formulas Γ.

We write $\Gamma \vdash \psi$ if the formula ψ is deducible from the set of formulas Γ in the first-order predicate calculus. We denote $\varphi \equiv \psi$ and say that formulas φ and

ψ are equivalent if $\varphi \vdash \psi$ and $\psi \vdash \varphi$. A deductively closed set of sentences $\mathbf{T}$ is called a theory: if $\mathbf{T} \vdash \psi$ implies $\psi \in \mathbf{T}$. If, for a set of sentences $\Gamma \subseteq S(\sigma)$, we have $\mathbf{T} = \{\psi \in S(\sigma) | \Gamma \vdash \psi\}$, we say that Γ is a set of axioms of theory $\mathbf{T}$. The set $Th(\Gamma) = \{\psi \in S(\sigma(\Gamma)) | \Gamma \vdash \psi\}$ is called a theory axiomatizable by the set of sentences Γ. In particular, $Th(\varphi) = Th(\{\varphi\})$ is a theory axiomatizable by the sentence φ. Symbol $\subset$ denotes strict inclusion: for sets A and B we have $A \subset B$ if $A \subseteq B$ and $A \neq B$.

Further, we state the basic ideas of the model-theoretical formalization of ontologies, developed by us [20–25].

For our approach, what information on a subject domain is contained in its ontology is most important. The basic content of subject domain ontology is the definition of the sense of key concepts of this subject domain. It means, in particular, that the information presented in the ontology should be valid in any instance of the described subject domain.

To determine what information on a subject domain should be contained in its ontology, we apply R. Carnap's approach to the classification of truth of sentences [2, 3]. R. Carnap considered three types of truth of sentences: logical truth, analytic truth and synthetic truth. A sentence is logical if the truth value of this statement depends only on its logical form. The sentence $(\varphi \vee \neg\varphi)$ is logically true, and the sentence $(\varphi \& \neg\varphi)$ is logically false. A sentence is analytic if its truth value depends only on the sense of concepts which are used in it. The sentence "a bachelor has not a wife" is analytically true, and "the dog is a bird" is analytically false. A sentence is synthetic if its truth value is determined by the conditions of the real world. The sentence "the Earth is a planet of the solar system" is synthetically true while the sentence "the Earth is flat" is synthetically false.

The content of a subject domain ontology is the definition of the sense of key concepts. Therefore, an ontology should contain only sentences which are analytic in the context of the given subject domain.

Notice that, from the point of view of our approach, a subject domain ontology contains only knowledge about the sense of concepts, in terms of which we speak about the given subject domain. Unlike ontology, the theory of the subject domain contains all the information about this subject domain. From our point of view, the distinction between subject domain ontology and the theory of this subject domain is identical with the distinction between analytic and synthetic sentences.

Thus, subject domain ontology should contain the set of key concepts of the subject domain and analytic sentences defining the sense of key concepts.

Definition 1. *The pair $O =< A, \sigma >$, where σ is the set of key concepts of a subject domain SD, and A is a set of analytic sentences describing the sense of these key concepts, is called a* ***formal ontology*** *of the subject domain SD. The set* ***T*** *of all sentences which are true in the subject domain SD, is called the theory of the subject domain SD.*

Note that $\sigma \subseteq \sigma(A)$, but $\sigma = \sigma(A)$ is not necessarily true. This means that the set of analytic sentences A can contain signature symbols which are not symbols

of key concepts of the subject domain. In the description of the sense of key concepts, it is possible to use sentences containing concepts which are not key concepts of the given subject domain.

Generally speaking, for a formal ontology $O =< A, \sigma >$, the set A is not a theory, that is, A is not deductively closed.

Definition 2. *Let the pair $< A, \sigma >$ be a formal ontology of a subject domain SD and let $\sigma_0 = \sigma(A)$. The set $\boldsymbol{T}_a = \{\varphi \in S(\sigma_0) | A \vdash \varphi\}$ is called the **analytical theory** of the subject domain SD.*

For the representation of knowledge contained in natural language texts, the notion of a linguistic network (see [27, 28] and [19, 20]) is effectively used. The linguistic network is a graph with vertices of two kinds: phrases of natural language and key concepts (words or word combinations). An edge of the graph connects two vertices if one of them is a phrase and the other is a key concept which is used in this phrase.

The linguistic network deals with concepts and sentences of natural language. For logical formalization of knowledge, it is necessary to pass from phrases and concepts of natural language to sentences of predicate logic and their signature. The formalization of the linguistic network is the *formal syntactic network*.

Definition 3. *Let σ be a signature and $\Gamma \subseteq S(\sigma)$. The triple $< \Gamma, \sigma, R >$ is called a **formal syntactic network** if $R \subseteq \Gamma \times \sigma$ and for any $\varphi \in \Gamma$ and $p \in \sigma$ the following holds:*

$$(\varphi, p) \in R \quad \Leftrightarrow \quad p \in \sigma(\varphi).$$

The weakness of the formal syntactic network is that signature symbols can appear in a sentence not only essentially, but also just formally. For example, signature symbols of a sentence ψ are included in the sentence $\varphi\&(\psi \vee \neg\psi)$ inessentially. To solve this problem, we introduce the notion of the *formal semantic network*.

Definition 4. *A formal syntactic network $< \Gamma, \sigma, R >$ is called a **formal semantic network** if for any $\varphi \in \Gamma$ and $p \in \sigma$ the statement $(\varphi, p) \in R$ implies that $p \in \sigma(\psi)$ for any sentence $\psi \equiv \varphi$.*

There is a natural question — whether arbitrary information can be represented by means of a formal semantic network? Can an arbitrary set of sentences of predicate logic be represented by a formal semantic network?

Denote by $[\varphi]_{\equiv}$ the set of all sentences (of some fixed signature) equivalent to the sentence φ.

Theorem 1. *For any set of sentences $\Gamma \subseteq S(\sigma)$ there exists a formal semantic network $< \Gamma_0, \sigma, R >$ such, that $\{[\varphi]_{\equiv} | \varphi \in \Gamma\} = \{[\varphi]_{\equiv} | \varphi \in \Gamma_0\}$.*

Theorem 1 implies that each set of sentences can be represented by a formal semantic network up to the equivalence of formulas. In particular, we can correctly represent any first-order theory by means of a formal semantic network.

Corollary 1. *Let T be a theory of a signature σ. There is a formal semantic network $< \Gamma, \sigma, R >$ which represents T, i.e.,*

$$\{[\varphi]_{\equiv} | \varphi \in T\} = \{[\varphi]_{\equiv} | \varphi \in \Gamma\}.$$

Definition 5. *Let T be a theory, $\sigma = \sigma(T)$, $\Gamma \subseteq S(\sigma)$ and Γ be a set of axioms of T, i.e., $T = \{\psi \in S(\sigma) | \Gamma \vdash \psi\}$. Γ is called the* ***canonical set of axioms*** *of theory T if*

a) for any sentences $\varphi \in \Gamma$ and ψ if $\varphi \equiv \psi$ then $\sigma(\varphi) \subseteq \sigma(\psi)$;

b) for any $\varphi \in \Gamma$ and any sentences ψ and ξ if $\varphi \equiv \psi \& \xi$ then $\sigma(\psi) \not\subset \sigma(\varphi)$ or $\sigma(\xi) \not\subset \sigma(\varphi)$.

Item (a) of Definition 5 means that each axiom belonging to the set Γ contains only those signature symbols which are used in this axiom essentially, and the signature of each axiom does not contain unnecessary garbage. Item (b) of Definition 5 demands the minimality of a set of concepts (signature symbols) of each axiom. It says that there are no sentences $\varphi \in \Gamma$, ψ and ξ such that $\varphi \equiv \psi \& \xi$, $\sigma(\psi) \subset \sigma(\varphi)$ and $\sigma(\xi) \subset \sigma(\varphi)$. Otherwise, in the set of axioms Γ instead of φ we should include sentences ψ and ξ each of which contains a smaller number of signature symbols than φ.

Theorem 2. *Any theory has a canonical set of axioms.*

Corollary 2. *For any theory, there is the greatest canonical set of axioms with respect to inclusion.*

Question 1. *Is there a minimal canonical set of axioms for each theory with respect to inclusion?*

This question has a positive answer for a special case which is very important in practice.

Corollary 3. *Every finitely axiomatizable theory has a minimal canonical set of axioms with respect to inclusion.*

Proposition 1. *There exists a theory which has no set of axioms Γ such that the set $\{[\varphi]_{\equiv} | \varphi \in \Gamma\}$ is least with respect to inclusion.*

In this connection, there is a similar question for finitely axiomatizable theories.

Question 2. *Is it true that each finitely axiomatizable theory has a canonical set of axioms Γ such that the set $\{[\varphi]_{\equiv} | \varphi \in \Gamma\}$ is least with respect to inclusion? Is it true that each finitely axiomatizable theory has a canonical set of axioms Γ which is minimal with respect to inclusion, such that the set $\{[\varphi]_{\equiv} | \varphi \in \Gamma\}$ is least with respect to inclusion?*

As we mentioned above, the main goal of ontology is the description of the sense of key terms of a subject domain. The most simple and widespread kind of definition of key terms is a glossary. A glossary is a sequence of definitions of some concepts which are, as a rule, key terms of a particular area of knowledge.

For the automation of the elaboration and use of glossaries, it is necessary to formalize the information contained in a glossary. We develop such formalization by means of first-order predicate logic.

Definition 6. *Let σ be a signature. The sequence of sentences $\varphi_1, \ldots, \varphi_n \in S(\sigma)$ is called a* ***formal glossary*** *(defining concepts from σ), if:*

a) $\sigma(\varphi_1) \subset \sigma(\varphi_1 \& \varphi_2) \subset \ldots \subset \sigma(\varphi_1 \& \ldots \& \varphi_n) = \sigma$*;*

b) $Th(\varphi_1 \& \ldots \& \varphi_k) = Th(\varphi_1 \& \ldots \& \varphi_n) \cap S(\sigma(\varphi_1 \& \ldots \& \varphi_k))$ *for every* $k \leq n$*, i.e., the addition of each new sentence φ_k conservatively extends the previous set of sentences.*

In each step of the glossary (that is, after each definition), some new sets of concepts (or, in the most simple case, one concept) is defined. New concepts are defined by means of the concepts defined earlier. The definition of these new concepts in the given step should be final and on further steps of the glossary this definition should not be changed. A concept defined once should not be redefined further. Otherwise, the whole of the glossary would be one huge definition of all key concepts, that is, concepts would be defined in only one step. Therefore, the conservatism of each step of a glossary is necessary.

As we have noted above, in the simplest case, in each step of the glossary, exactly one new concept is defined. Such definitions of a glossary may be explicit or implicit.

Definition 7. *We say that a formal glossary $\varphi_1, \ldots, \varphi_n$* ***explicitly defines*** *concepts from σ if there are formulas $\psi_1, \ldots, \psi_n$ such that for every $k < n$ one of the following conditions holds:*

$$\varphi_{k+1} = \forall \overline{x}(P(\overline{x}) \leftrightarrow \psi_{k+1}(\overline{x})),$$

$$\varphi_{k+1} = \forall \overline{x} \forall y((f(\overline{x}) = y) \leftrightarrow \psi_{k+1}(\overline{x}, y)),$$

$$\varphi_{k+1} = \forall y((c = y) \leftrightarrow \psi_{k+1}(y)),$$

where $P, f, c \in \sigma \backslash \sigma(\varphi_1 \& \ldots \& \varphi_k)$, $\overline{x}$ is a tuple of variables and $\sigma(\psi_{k+1}) \subseteq \sigma(\varphi_1 \& \ldots \& \varphi_k)$.

These three kinds of explicit definitions are definitions of signature predicate, function and constant.

Question 3. *Can the sense of key concepts always be represented by an explicit glossary — a sequence of explicit definitions?*

Notice that, in the case of an explicit glossary, each new definition contains exactly one new signature symbol.

Definition 8. *We say that a formal glossary $\varphi_1, \ldots, \varphi_n$ represents a set of sentences Γ if $Th(\Gamma) = Th(\varphi_1 \& \ldots \& \varphi_n)$.*

Remark 1. *There exists a formal glossary which represents a set of sentences S if and only if theory $Th(S)$ is finitely axiomatizable.*

Question 4. *Can each finitely axiomatizable theory be represented by an explicit glossary?*

Remark 2. *For any set of sentences Γ of a finite signature σ there exists a conservative sequence of sets of sentences $\Gamma_1, \ldots, \Gamma_n$ such that $Th(\Gamma_1 \cup \ldots \cup \Gamma_n) = Th(\Gamma)$, $\sigma(\Gamma_1) \subset \sigma(\Gamma_1 \cup \Gamma_2) \subset \ldots \subset \sigma(\Gamma_1 \cup \ldots \cup \Gamma_n)$ and for any $k < n$ the set $\sigma(\Gamma_1 \cup \ldots \cup \Gamma_{k+1}) \backslash \sigma(\Gamma_1 \cup \ldots \cup \Gamma_k)$ contains exactly one symbol, and also the set $\sigma(\Gamma_1)$ contains exactly one symbol.*

Remark 2 shows that, by means of a conservative sequence of infinite sets of sentences, we can always represent definitions of key concepts so that in each step exactly one new concept would be defined. Definitions of real glossaries contain only a finite number of sentences. Therefore, there is a question — whether it is always possible to organize a glossary so that concepts would be defined one by one? For real glossaries written in natural language, we have just this situation. The question is — whether it is always possible, for any definitions of key concepts?

Question 5. a) *If the set of sentences Γ is finite, is it possible in Remark 2 to find a sequence of sets of sentences $\Gamma_1, \ldots, \Gamma_n$ such that all theories $Th(\Gamma_k)$ would be finitely axiomatizable?*

b) *Is it true that for any sentence φ there exists a formal glossary $\varphi_1, \ldots, \varphi_n$ such that $Th(\varphi) = Th(\varphi_1 \& \ldots \& \varphi_n)$ and for each $k < n$ the set*

$$\sigma(\varphi_1 \& \ldots \& \varphi_{k+1}) \backslash \sigma(\varphi_1 \& \ldots \& \varphi_k)$$

would contain exactly one signature symbol, and also the set $\sigma(\varphi_1)$ would contain exactly one symbol?

Theorem 3. *There is a signature $\sigma = \{s_1, s_2\}$ and a sentence φ defining concepts from σ, for which there exists no formal glossary φ_1, φ_2 representing $\{\varphi\}$ such that $\emptyset \neq \sigma(\varphi_1) \subset \sigma(\varphi_2)$.*

The negative answer to Question 5 follows from Theorem 3.

Corollary 4. *In the general case, definitions of concepts cannot be represented in the form of a glossary defining concepts one by one.*

Corollary 5. *In the general case, the sense of concepts cannot be represented in the form of an explicit glossary. In particular, it is not true that each finitely axiomatizable theory can be represented by an explicit glossary.*

Corollary 5 gives a negative answer to Questions 3 and 4.

Proof of Theorem 3. To prove the theorem, we consider the theory of Boolean algebras with distinguished ideals [15–18]. We show that the answers to Questions 3–5 are negative even for a definition of two concepts presented by unary predicate and binary predicate in the context of the theory of Boolean algebras with distinguished ideals.

Consider the signature $\sigma = \{\leq, I\}$ consisting of a symbol of binary predicate $\leq$ and a symbol of unary predicate I.

Lemma 1. *There exists a sentence φ of first-order predicate logic of the signature σ, stating that each model on which this sentence is true satisfies the following conditions:*

1. *this model is a Boolean algebra naturally ordered by the relation $\leq$;*
2. *this Boolean algebra is atomic;*
3. *the unary predicate I distinguishes an ideal of this Boolean algebra;*
4. *each atom of the Boolean algebra belongs to the ideal I;*
5. *the unit of the Boolean algebra does not belong to the ideal I.*

Let us consider the sentence φ introduced in Lemma 1.

Lemma 2. *Each model of the sentence φ is an infinite atomic Boolean algebra.*

Lemma 3. *For any model of the sentence φ, both predicate I and its complement contain an infinite number of elements.*

Remark 3. *The theory of the class of infinite atomic Boolean algebras is not finitely axiomatizable.*

Denote $\sigma_1 = \{\leq\}$ and $\sigma_2 = \{I\}$.

Corollary 6. *The theory $Th(\varphi) \cap S(\sigma_1)$ is not finitely axiomatizable.*

Remark 4. *The theory of the class $\boldsymbol{K}$ of models of the signature $\{I\}$ consisting of one unary predicate, in which both the predicate and its complement contain an infinite number of elements, is not finitely axiomatizable.*

Corollary 7. *The theory $Th(\varphi) \cap S(\sigma_2)$ is not finitely axiomatizable.*

Proposition 2. *For the sentence φ there is no formal glossary φ_1, φ_2 representing $\{\varphi\}$, such that $\emptyset \neq \sigma(\varphi_1) \subset \sigma(\varphi_2)$.*

Theorem 3 follows from Proposition 2.

Thus, in this section, we introduced a natural internal structure on sentences of first-order predicate logic — the formal semantic network. This internal structure can be used, in particular, for metrics of proximity between key concepts which are formally presented as signature symbols. The metrics of proximity between key concepts are based on the length of the shortest way between two vertices of a formal semantic network: if the length is larger, then the proximity of two given concepts is less.

With the help of such a measure of proximity, we determine concepts associated with a given concept. In particular, by the name of a subject domain, we determine concepts which are most typical for this subject domain.

By means of model theory, we formalized the notion of a glossary and investigated the possibility of representation of the sense of key concepts by an explicit glossary.

The developed mathematical tools enable us to make a formal representation of ontological knowledge, which is a basis of the achievement of relevance of information retrieval by the Virtual Catalog.

4 Practical Realization. The Virtual Catalog

4.1 Description of the Virtual Catalog

Remember the requirements for search systems, which we have formulated above, in Section 2.1.

1. Great expressive force: the ability to represent various information needs.
2. Clarity for the user. The sense of a created formal query should be completely clear to the user and should not allow ambiguities.
3. The creation of a formal query should not take too much time.
4. The use of tools for the creation of a formal query should not demand special training or education.

These requirements underlie the metasearch system developed by us — the Virtual Catalog.

As we mentioned above, the first item of these requirements is in certain contradiction with the remainder, especially with items 3 and 4. Nevertheless, this contradiction is solvable. Fast and clear specification of search tasks can be achieved if we decompose the search task into a composition of subtasks. For this purpose, it is necessary to introduce dimensions of a search task.

We distinguish three dimensions of a search task:

– the subject domain in which the information is searched;
– the kind of desired Internet resource;
– the kind of search problem.

Thus, a search task, which the user should specify, is represented in the form of a triple of values. Its components are the subject domain, the kind of Internet resource and the kind of search problem.

With each dimension of the search task, we connect an ontology. For the realization of an information search, we need three kinds of ontology: subject domain ontology, ontology of the Internet and ontology of search tasks.

Accordingly, the interface of the Virtual Catalog consists of three parts:

1. Catalog of subject domains.
2. Catalog of kinds of Internet resources.
3. Catalog of search tasks.

The catalog of subject domains is similar to the interface of a usual Internet catalog. There is a structure of headings and subheadings, having several levels of nesting. The Virtual Catalog is intended for information searching in some fields of science — in mathematics, chemistry, patent branches and so on. Therefore, the names of the headings correspond to subfields in this field of science — for example, to areas of mathematics.

There are two essential differences between the catalog of subject domains and usual Internet catalogs.

First, the hierarchy of headings is not a tree. A heading can be a direct subheading of several headings, instead of one, as it is usually accepted.

Secondly, the user can choose a heading from any level of hierarchy of nesting and start the search. The reply will contain documents corresponding to the specified heading of the given level of the hierarchy of nesting.

Such an approach to organize the structure of headings eliminates two problems. On the one hand, it is not necessary for developers of the system of headings to take trouble over the question of with what heading the given subheading should be connected; often it is not so obvious. On the other hand, we get rid of one of the widespread inconveniences of interfaces of program systems, arranged in the form of a tree. In such interfaces, to find the desired final heading, the user sometimes needs to scan several subtrees of headings, which may contain the given final heading.

Unlike a usual Internet catalog, the Virtual Catalog does not contain any information about concrete Internet resources. It works as a metasearch system. By the triple choice made by the user — subject domain, kind of Internet resource and kind of search task — the Virtual Catalog generates a query to one or several search systems (depending on the user's query) and receives a reply from these search systems. Further, this reply is processed and given out to the user.

The main idea of the Virtual Catalog is to automate routine work which the user carries out each time to find the desired information by means of search engines, such as Google, Yahoo and AltaVista. Such automation reduces the time expenses of the user and improves the quality of the found Internet resources. Not every Internet user is experienced in work with search systems. The Virtual Catalog creates such a query to the search engine, which only a very qualified user can make, having spent plenty of time and many attempts.

For the successful realization of the Virtual Catalog, it is necessary for us:

– to develop a classification of the search tasks which are represented as a composition of three components;
– to develop methods of generating formal queries for each search task.

Thus, to develop convenient and effective tools of information retrieval, we should solve the following problems:

1. Description of the components which determine the search task.
2. Classification of search tasks.
3. Classification of types of users by search tasks.
4. Clustering of search problems; division of the problems into classes.
5. Development of schemes of specification of search problems of the given user.

To determine components which specify the search problem, that is, to decompose the search problem into subproblems, we use the methodology of the speech act theory [19, 20].

The development of the Virtual Catalog enables us to satisfy all the requirements of the search system formulated in this section. Thus, high efficiency of the information search will be achieved.

The adequacy of the formal query is achieved by means of tools of formulation of the search query and by clarity of these means for the user. The relevance

of working of the formal query is achieved by the use of three ontologies corresponding to the components of the formal query — ontology of subject domain, ontology of Internet resources and ontology of search tasks. As a result, we have high pertinence of information retrieval by the Virtual Catalog.

4.2 Automation of the Development of Subject Domain Ontologies

In the third paragraph, we have stated mathematical methods of representation of ontological knowledge on the given subject domain. In this section, we state practical technologies of the creation of ontologies, which we use for the development of the Virtual Catalog.

As we have mentioned above, for the realization of the Virtual Catalog, we need three kinds of ontology — subject domain ontology, ontology of types of Internet resources and ontology of search tasks. In the first version of the Virtual Catalog, the main attention is given to the hierarchy of ontologies of subject domains.

There are two cardinal ways of obtaining essential knowledge about a subject domain, which may be presented in the form of an ontology of this subject domain:

– extraction of ontological knowledge from experts in the subject domain;
– extraction of ontological knowledge from natural language texts describing this subject domain.

The weakness of the first method is high cost and the need of plenty of time. Using such a method for the development of high-quality ontologies, the participation of high class experts on this subject domain is necessary. An attempt for a completely manual development of ontologies would be rather expensive. Therefore, we use the second method; we take ontological information on the subject domain from natural language texts.

There is an opportunity to use the work on defining the sense of key concepts, which has already been done. We can use encyclopaedic dictionaries related to the given area of knowledge. A collective of leading experts in the given area, as a rule, takes part in the elaboration of such dictionaries. Therefore, the information contained in such texts is canonical for specialists in the given area.

In this case, the problem consists of extracting ontological knowledge from encyclopaedic dictionaries. First, it is necessary to convert the text of the dictionary to a structured form (for example, to an XML document). Secondly, it is necessary to extract ontological information from the structured text.

To solve these problems, we need automation of the extraction of ontological information from natural language texts. For the elaboration of the Virtual Catalog, we have a special project devoted to the development of software for parsing natural language texts.

4.3 Elaboration of Virtual Catalogs for Mathematics and Information Security

Now, we develop Virtual Catalogs for two areas — mathematics and information security. Also, we start to elaborate a Virtual Catalog for the patent branch.

The purpose of the Virtual Catalog is to solve any problem concerning a search for scientific and technical information in the given area — mathematics or information security. The development of the Virtual Catalog for mathematics is connected with the project MathTree, "The Treelike Catalog of Mathematical Internet Resources" (www.mathtree.ru).

The Virtual Catalog is a metasearch system which processes the formal query of the user and generates queries to search engines — universal, such as Google and Yandex, and also to specialized search engines. As a result, the user receives a reply which is pertinent to his formal query.

As we already mentioned, the search problem which the user should formalize is represented in the form of a three-dimensional vector. The first component is a field of a subject domain. The second component is a type of Internet resource. The third component is a type of search task.

Now we have realized the first two components — the field of the subject domain and the type of Internet resource. In the future, the type of search problem will also be realized.

The specification of a field of a subject domain is organized in the form of a catalog: there is a rubricator with several levels of nesting — up to five levels. This catalog has two features.

First, the user can choose a heading from any level of nesting, instead of necessarily the final one. For example, the user can choose "Algebra and logic", "Algebra" or "Group theory". Then the search will be executed throughout the entire specified field of science.

Secondly, headings are structured by the relation "is a kind of" (heading–subheading); however, this relation is not a tree. A heading of any level of nesting can be a direct subheading of one heading as well as of several headings.

Such a system of headings in fields of science, used in the Virtual Catalog, differs from almost all search systems on scientific and technical information.

At the present time, the following kinds of Internet resources are implemented: "articles", "electronic editions", "sites of magazines", "electronic libraries", "scientific communities", "scientific schools", "conferences", "organizations", "dissertational councils", "forums" and "personal pages".

The user can add a few keywords to the formal query generated by the system; however, it is not necessary. The keywords added by the user will be included in the query to the search engine together with heuristics which are generated by the Virtual Catalog. The adding of keywords gives to an experienced user an opportunity, first, to make the query more exact and, second, to carry out customizing and fine-tuning of the system for the user and his vision of the subject domain and types of Internet resources. If the result obtained by adding new keywords is good for the user, it may be kept, and further it will be used automatically.

Elaboration: the expansion and specification of the list of types of Internet resources is a research problem which we investigate. In the future, we are going to use an essentially more detailed set of types of Internet resources, which will be presented in the form of a catalog with headings and subheadings.

For relevant processing of the formal query specified by the user, in the Virtual Catalog the function for the generation of heuristics is implemented. It is a binary function; the first argument is the chosen heading of the catalog, and the second argument is a type of Internet resource. By the given pair of arguments, the Virtual Catalog generates the heuristics set. On the base of this set, the Virtual Catalog creates the query to the search engines.

For high relevance of information retrieval, the selection of proper heuristics sets is very important. For this purpose, the experimental machine for selecting heuristics is developed.

The interface of the Virtual Catalog allows working with it at two levels: as a user and as an expert-developer. An expert has special means of selecting heuristics and comparing their efficiency, and means of updating and changing the heuristics set for each pair - a heading of a subject domain and a type of Internet resource. As a result of the work of experts, there is a constant increase of relevance of processing search queries by the Virtual Catalog. In the next paper we will consider the procedure of generating heuristics sets in details.

Thus, because of the representation of a formal query as a three-dimensional vector, great adequacy of the formal query is achieved. Due to the heuristics selection, we have good relevance of processing the formal query. As a result, we attain high pertinence of the Virtual Catalog.

5 Conclusion

At the present time, searching for various information contained on the Internet is an art. Different people have different search skills and abilities, and each expert in information retrieval has his own tricks (corresponding to the set of search problems usually solved by him). The purpose of our research is, first, to transform this art into technology; secondly, this technology should be automated; and, thirdly, it is necessary to determine classes of search tasks and relate search technologies to these classes.

To solve these problems, we developed the Virtual Catalog — the metasearch system intended for the retrieval of scientific and technical information on the Internet.

The elaboration of the Virtual Catalog is based on the formal representation of subject domain ontologies in the language of model theory. In the paper, the basic notions and results of the model-theoretical approach to the formalization of ontologies are presented.

In the Virtual Catalog, the use of subject domain ontologies, ontologies of Internet resources and ontologies of search problems gives an opportunity to formulate the informational need of the user more precisely and completely. The module of generating heuristics sets provides relevance of processing the formalized query. As a result, the Virtual Catalog achieves high pertinence of information retrieval.

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, Cambridge (2003)
2. Carnap, R.: Meaning and Necessity: A Study in Semantics and Modal Logic. The University of Chicago Press, Chicago (1947)
3. Carnap, R.: Philosophical Foundations of Physics. Basic Books, New York (1968)
4. Chang, C.C., Keisler, H.J.: Model Theory. North-Holland Publishing Company, New York (1973)
5. Daconta, M.C., Obrst, L.J., Smith, K.T.: The Semantic Web. In: A Guide to the Future of XML, Web Services, and Knowledge Management, Wiley Technology Publishing, Indianapolis (2006)
6. Ershov, Y.L., Palutin, E.A.: Mathematical Logic. Walter de Gruyter, Berlin (1989)
7. Fensel, D.: Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce, 2nd edn. Springer, Berlin (2003)
8. Gangemi, A., Pisanelli, D.M., Steve, G.: An overview on the ONIONS project: Applying ontologies to the integration of medical terminologies. Data and Knowledge Engineering 31(2), 183–220 (1999)
9. Gómez-Pérez, A., Fernandez-Lopez, M., Corcho, O.: Ontological Engineering with Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web. Springer, London (2004)
10. Haarslev, V., Möller, R.: RACER System Description. In: Goré, R.P., Leitsch, A., Nipkow, T. (eds.) IJCAR 2001. LNCS (LNAI), vol. 2083, pp. 701–705. Springer, Heidelberg (2001)
11. Haarslev, V., Möller, R.: RACER: An OWL Reasoning Agent for the Semantic Web. In: Proceedings of the International Workshop on Applications, Products and Services of Web-based Support Systems, in Conjunction with the 2003 IEEE/WIC International Conference on Web Intelligence, Halifax, Canada, pp. 91–95 (2003)
12. Maedche, A.: Ontology Learning for the Semantic Web. The Kluwer International Series in Engineering and Computer Science, vol. 665 (2003)
13. McGuinness, D.L., van Harmelen, F.: OWL Web Ontology Language Overview. Recommendation, World Wide Web Consortium (2004), `http://www.w3.org/TR/owl-features/`
14. Mizoguchi, R.: Ontological Engineering: Foundation of the Next Generation Knowledge Processing. In: Zhong, N., Yao, Y., Ohsuga, S., Liu, J. (eds.) WI 2001. LNCS (LNAI), vol. 2198, pp. 44–57. Springer, Heidelberg (2001)
15. Palchunov, D.E.: Countably-categorical Boolean algebras with distinguished ideals. Studia Logica XLVI(2), 121–135 (1987)
16. Palchunov, D.E.: Finitely axiomatizable Boolean algebras with distinguished ideals. Algebra and Logic 26(4), 252–266 (1987)
17. Palchunov, D.E.: Direct summands of Boolean algebras with distinguished ideals. Algebra and Logic 31(5), 295–316 (1992)
18. Palchunov, D.E.: The Lindenbaum-Tarski algebra for Boolean algebras with distinguished ideals. Algebra and Logic 34(1), 50–65 (1995)
19. Palchunov, D.E.: Algebraische Beschreibung der Bedeutung von Äußerungen der natürlichen Sprache. In: Zelger, J., Maier, M. (Hrsg.) GABEK. Verarbeitung und Darstellung von Wissen, pp. 310–326. STUDIENVerlag, Innsbruck (1999)
20. Palchunov, D.E.: On a logical analysis of GABEK. In: Buber, R., Zelger, J. (Hrsg.) GABEK II. Zur Qualitativen Forschung. On Qualitative Research, pp. 185–203. STUDIENVerlag, Innsbruck (2000)

21. Palchunov, D.E.: Modelling of thinking and formalization of reflection I: Model theoretical formalization of ontology and reflection. Philosophy of Science 31(4), 86–114 (2006) (in Russian)
22. Palchunov, D.E.: GABEK for Ontology Generation. In: Herdina, P., Oberprantacher, A., Zelger, J. (eds.) Lernen und Entwicklung in Organisationen. Learning and Development in Organizations, Beitrage zur Wissensverarbeitung, 2nd edn., Berlin, Wien (LIT), pp. 90–109 (2007)
23. Palchunov, D.E., Sidorova, E.S.: The Virtual Catalog. In: Proc. All-Russia Conference "Knowledge–Ontology–Theory", Novosibirsk, pp. 166–175 (2007) (in Russian)
24. Palchunov, D.E.: Solution of the problem of information retrieval, based on ontologies. Business Informatics (1), 3–13 (2008) (in Russian)
25. Palchunov, D.E.: Definability of sentences in the language of Boolean algebras with distinguished ideals. Vestnik NSU. Mathematics, Mechanics, Informatics 8(2), 92–105 (2008) (in Russian)
26. Staab, S., Studer, R. (eds.): Handbook on Ontologies. Springer, Heidelberg (2004)
27. Zelger, J.: GABEK, a new method for qualitative evaluation of interviews and model construction with PC-support. In: Stuhler, E., Suilleabhain, M.O. (eds.) Enchanging Human Capacity to Solve Ecological and Socio-Economic Problems, pp. 128–172. Rainer Hampp Verlag, Munchen (1993)
28. Zelger, J.: Zur Geschichte von GABEK. In: Buber, R., Zelger, J. (Hrsg.) GABEK II. Zur Qualitativen Forschung. On Qualitative Research, pp. 13–20. STUDIENVerlag, Innsbruck (2000)

Ontology Development for Domains with Complicated Structures

Irina L. Artemieva

Institute for Automation & Control Processes
The Far-Eastern Branch of the Russian Academy of Sciences,
Radio str. 5, 690041 Vladivostok, Russia
artemeva@iacp.dvo.ru
http://www.iacp.dvo.ru/is/

Abstract. The importance of ontology is generally recognized today: as the base for specification and development of software, shared information access, knowledge portal development, user interfaces of software and information editors. However, existing ontology descriptions and their development methods do not embrace complicatedly structured domains: domains with different but similar subdomain ontologies, subdomains with different but similar sub-subdomain ontologies and so on. This paper contains a description of the class of complicatedly structured domains and provides examples. The definition of multilevel ontologies for such domains is described; the method of their development is presented. The differences between this new method and already existing methods for ontology creation are analyzed. The properties of intelligent systems for complicatedly structured domains based on multilevel ontologies are considered.

Keywords: Domains with complicated structures, multilevel ontologies, methodologies for multilevel ontology development.

1 Introduction

At present using ontologies as bases for specification and development of software [1], shared information access, information search, information merging interaction, knowledge portal development [2,3,4], user interfaces of software [5] and information editors [6,7] is considered to be important. Many studies are aimed at developing exact formal knowledge catalogues that can be used by intelligent systems. Many works consider the ontology as a formal explicit description of concepts (often called classes) of a domain, properties (sometimes called slots) of each concept. The description assigns various features and attributes of a concept, and restrictions on properties (sometimes called facets) [8,9]. The ontology defines terms used for description and representation of domain knowledge. The ontology together with a set of individual instances of classes constitutes a knowledge base.

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 184–202, 2011.

By now, different methodologies for making ontologies [10,11,12] for the above applications have been developed. In many methodologies the ontology development includes (1) definition of classes in the ontology; (2) arrangement of classes in a taxonomic hierarchy (subclass – superclass); (3) definition of slots and description of allowed values for these slots; (4) filling in the values for slots for instances [13]. These methodologies were used to develop different ontologies (e.g. `http://musing.deri.at/ontologies/v0.3/` and `http://www.daml.org/ontologies/`) to be used by program systems of the above classes.

However, some problems exists. The above definitions of ontologies do not embrace complicatedly structured domains, i.e. domains with different but similar [15] subdomain ontologies, subdomains with different but similar sub- subdomain ontologies and so on. Top-level and upper-level ontologies [14] contain definitions of high-level abstraction concepts, which make it difficult to use them to model domains. It is often that the level, understood as an ontology level, is a characteristic of a concept of an ontology and defines the level of a class in the class hierarchy. Ontologies as concept systems that describe domain knowledge can be useful to develop systems for knowledge storage but such ontologies lack concepts that can be used to define input data and results of applied task solutions except for tasks of structuring, editing and searching for information. Also, there are no methods for using ontologies to develop program systems that solve applied tasks except for tasks of structuring, editing and searching for information.

The aim of this article is describing the class of domains with complicated structures, components of their ontologies and the method of developing them. The differences between this new method and already existing methods for ontology creation are analyzed.

2 Domain Class Definition

Domains with complicated structures have the following characteristics:

- they have sections that are described in different but similar [15] concept systems;
- sections have subsections that are described in different but similar concept systems;
- any subsection can have subsections with the above characteristic.

Sections (and subsections) of domains with complicated structures are also domains with their own kinds of activity and sets of applied tasks; some applied tasks from different sections may be similar. Concepts of ontologies and knowledge of different sections can be used to solve applied tasks in domains with complicated structures.

Chemistry is an example of a domain with a complicated structure. The examples of its sections are physical chemistry, organic chemistry and analytical chemistry. Physical chemistry deals with physicochemical processes [16]. These processes are described in terms of characteristics of substances and reactions that take part in the processes. Organic chemistry adds terms relating to

structural properties of substances [17,18]. Analytical chemistry studies processes of influence on substances with various kinds of radiation [19]. Chemical thermodynamics and chemical kinetics are examples of sections of physical chemistry; sections of analytical chemistry depend on analysis techniques (for example, X-ray fluorescence analysis).

Another example of a domain with a complicated structure is the program transformations domain. It studies the processes of changing programs as a result of applying different transformations [20]. Transformations of structural programs and transformations of parallel programs are examples of the sections. Transformations are described in terms of properties of languages of these programs.

Diagnosing, designing, planning, etc. are examples of tasks solved in domains with complicated structures. Terms that describe characteristics of substances used in an experiment, reactions that occur over its course, its chemical process conditions [21] are necessary for specifying an applied task of planning a chemical experiment. These terms differ from terms used to describe characteristics of substances and reactions that are traditionally stored in chemistry databases.

Some problems arise when intelligent systems for domains with complicated structures are designed. The first problem is integration of knowledge from various sections and subsections within the framework of one knowledge base. A means of such integration is ontology that must take into account that concept systems (ontologies of sections and subsections) used in different sections and subsections differ. The second problem is a way of integration of concept systems (ontologies). A means of such integration can be ontology of higher level of generality.

3 Defining Level of Generality of Ontologies

Let us define a concept "level of generality of ontologies" [22]. Ontologies are used to verbally represent information. Verbal representation of information is a mapping of a finite set of terms into a set of possible values of terms. Verbal representation of information has level 0. Ontology with the system of knowledge that specifies a particular set of verbal representations of information has level 1. At level 1 ontology terms have no values. Setting values to ontology terms make the verbal representation of the particular information have level 0.

Ontology without knowledge system has level 2. When different knowledge is added, different specifications of verbal representations of information are received.

Ontology in terms of which ontology of level i can be specified has level i+1. All ontologies of levels more than 2 are metaontologies.

Let us explain the difference between level 1 and level 2. Ontology defines a set of terms used to verbally represent information, a set of possible values of each term, and relations between terms (ontological constraints). Ontology is the result of agreement among people who use the same information in their

discourse; therefore ontology is obviously the result of agreement among these people on what verbal representations in the domain have meaning. Let us use the term "conceptualization" for a set of all verbal representations of information that have meaning. Knowledge imposes additional constraints on a set of verbal representations and picks a subset out of conceptualization. Thus, level 2 specifies conceptualization, level 1 specifies its subset, where knowledge defines characteristics of this subset.

Ontology of the next level specifies a larger set of verbal representations of information as compared with ontology of the previous level. Transition from level i to level $i-1$ restricts this set and defines its subset. Transition from ontology of level $i-1$ to ontology of level i is considering ontology of level $i-1$ as verbalized information.

Let us consider domains as examples of verbalized information. Knowledge is represented verbally if it is specified as an array of pairs consisting of a term and its value. Terms included into the ontology of knowledge is used to verbally represent knowledge [23]. If verbalized information is a knowledge base of a domain, then its representation has level 0. If this is the case, level 1 is ontology of knowledge consisting of definitions of terms with their sets of values and knowledge consistency constraints as well. Level 2 specifies sets of ontologies of knowledge.

Let us consider the example when information is a description of a state of affairs of the domain [23]. In respect of physical chemistry, level 0 represents the information of a certain physicochemical process that took place at a certain period of time and under certain external conditions. To describe the process one may use the terms with the following values: "process steps", "chemical substances at each process step", "chemical reactions at each process step", etc. The terms used for describing level 0 for the domain form the ontology of reality. Level 1 specifies the reality model of the given domain and describes all possible chemical processes the information about which can be represented in terms of the ontology; the representation of the information about each chemical process is not inconsistent with the ontological constraints and domain knowledge. The domain knowledge describes laws for going of chemical reactions, formation laws for substance from chemical elements, etc. Level 2 specifies the conceptualization of reality that is an idea about reality that the domain specialist has. This level defines concepts for this reality description.

Regarding X-ray fluorescence analysis, a section of analytical chemistry, level 0 represents the information about a certain physical process that took place at a certain period of time and under certain external conditions. To describe the process one may use the terms with the following values: "analytes", "sample qualitative composition", "percentage of an analyte in a sample" [19]. The domain knowledge describes laws of physical processes during high-frequency electromagnetic radiation directed at a sample, values of characteristic radiation of analytes, etc.

The domain knowledge defines characteristics of its reality as sets of states of affairs that can take place in it. If the domain knowledge can be verbally

represented, then the ontology of level 2 contains sets of terms for their representation. This ontology is a pair of two ontologies (reality and knowledge) and relations between them (additional ontological constraints). If the domain knowledge cannot be verbally represented (e.g. physics), then the ontology of level 2 coincides with the ontology of reality. In this case, the ontology of level 3 specifies a set of ontologies of reality.

There are domains where only part of knowledge is verbally represented. In this case, the ontology of knowledge contains terms that can represent this knowledge. This is characteristic of physical chemistry: knowledge about various properties of chemical elements (atomic weight, atomic number, etc.), physicochemical characteristics of substances (density, formula, etc.), reaction properties (e.g. catalyst) and so on is verbally represented in it. Laws of physical processes cannot be represented verbally.

4 Properties of Multilevel Ontologies for Domains with Complicated Structures

For a domain with a complicated structure, the level with the maximum number n is an ontology of the domain (Fig. 1). The level contains the terms with the help of which the ontology of the next level is defined. The transition to the next level means specifying ontology terms and ontological constraints of the next level [22]. If all the domain knowledge and ontological constraints of all the levels can be represented verbally, then the ontology of level n defines properties of all sets of terms of all ontologies of lower levels. The simplified ontology of medical diagnosis has this property [23,24].

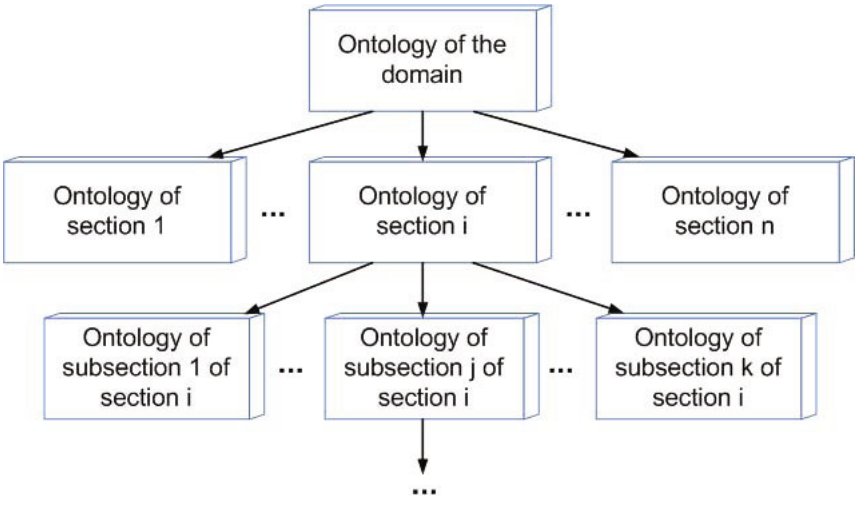


Fig. 1. An ontology construction for a domain with a complicated structure

The ontology of level $n-1$ consists of modules. Each module defines the ontology of a certain section of the domain. The ontology of level $n-2$ also consists of modules. Each module defines the ontology of a subsection. All the ontologies of level lower than n are modular. The domain knowledge base is also modular. Each module of knowledge consists of knowledge of a subsection.

5 Method of Developing Ontology for Domain with Complicated Structure

Let us describe the method for developing a multilevel ontology of a domain with a complicated structure.

The development starts with defining verbalized information about the domain reality and terms for its verbal representation [1]. The domain expert participates in this work. The knowledge engineer and the expert make a list of terms used for representing reality, record meanings of terms and values, principles of representing states of affairs with their help. A set of all possible meanings is defined for each term (denotation of the term). Ontological agreements specifying constraints for a set of meanings of terms are formed (domain state of affairs consistency constraints).

A set of applied tasks of a discourse is analyzed in order to define what information about the reality is to be verbally represented. Terms used for specifying input data and their results and terms used for representing values of intermediate data are defined. Ontological agreements specifying relations between all these terms are also defined. A set of tasks of the discourse unambiguously defines the domain. Thus, the ontology of the reality contains terms that are used in applied tasks to specify input data and results of solutions, intermediate data, and relations between terms of the three groups. Then, the knowledge system, probably defining the reality of the domain more accurately, is to be designed.

Developing the ontology of level 2 starts with answering the question whether the domain knowledge can be represented verbally. If the answer is in the negative (i.e. the knowledge cannot be represented verbally), the knowledge system is developed in the same form as the system of ontological constraints (in terms of the ontology of the reality). If the answer is in the affirmative (i.e. the knowledge can be represented verbally), a list of terms for representing the domain knowledge is made with the help of an expert, definitions of these terms are developed, knowledge consistency constraints and interrelations between the reality and knowledge are formulated. This list of terms for representing the domain knowledge and the set of knowledge consistency constraints form the ontology of this domain knowledge. If only part of knowledge can be represented verbally, a list of terms for representing only this part is made. The system of knowledge consists of two components: a set of assertions in terms of the ontology of reality and mapping of a set of terms for representing knowledge into a set of values. The ontology of the reality, the domain knowledge ontology, and interrelations between the reality and knowledge form the ontology of level 2 for the domain.

The ontology of level 3 (and all the following levels) can be developed if the ontologies of level 2 (and all the previous levels) of several sections of the domain are developed since this ontology specifies a set of ontologies of level 2. This step starts with answering whether the ontologies of level 2 can be represented verbally. If the answer is in the negative, developing stops. If the answer is in the affirmative, a list of terms for its representing is made, definitions of these terms are developed, consistency constraints for the ontology of level 2 are formulated. The task of this level and the following ones is searching for "regularities" in the ontology of the previous level. To do it, terms with some similar sense are grouped into one set, a name for each set is defined, the common sense of terms from a set is defined for each set, and relations among these sets are described. Also, ontological constraints with some similar sense are grouped into one set, the common sense of constraints of a set is formulated for each set, and relations among these sets are defined.

6 Structure of Multilevel Ontology of Chemistry

When describing a multilevel ontology of chemistry, I assume that ontologies of level 2 have been designed (you can find all of them at `http://www.iacp.dvo.ru/is/`). The structure of the reality and the structure of the knowledge of chemistry sections were analyzed. As it is mentioned above, developing ontologies of level 3 and other following levels requires search for "regularities" in ontologies. Let us exemplify the method of developing the ontology of level 3.

Terms (names of chemical element properties) can be specified in the ontology of knowledge of physical chemistry. Each such term means the function the argument of which is a chemical element, and the result of which is a value of the property, i.e. the denotation is a mapping the range of definition of which is a set of chemical elements and the range of values is owned by each property. The same terms can be found both in the ontology of organic chemistry and in ontologies of other sections. Apart from terms that define names of properties of elements, there are also terms that are names of properties of chemical compounds, reactions, radicals, etc.

The analysis of the mentioned ontologies shows that there are other sets of terms that share some common properties. Such terms are terms that make it possible to identify compositions of chemical elements or chemical substances. Each such term is the function the argument of which is a chemical element or substance, and the result of which is a set of components of an element or substance. There are also terms that define various properties of an element in the composition of a substance, a substance as a reactant, etc.

For the above examples, the ontology of level 3 contains the following terms: (i) "own properties of elements" (a set of functions the range of definition of which is a set of chemical elements and the range of values depends on the term that assigns a name of a property); (ii) "own properties of substances" (a set of functions the range of definition of which is a set of chemical substances and the range of values depends on the term that assigns a name of a property);

(iii) "components of substance" (a set of functions the range of definition of which is a set of chemical substances and the range of values depends on the type of a component; it can be, for example, a set of chemical elements or functional groups); (iv) "reactants" (a set of functions the range of definition of which is a set of chemical reactions and the range of values depends on the way of considering a reaction; it can be, for example, a set of chemical substances), etc.

Transition to the next level (4) takes place in the following way: if in the ontology of level 3 there are terms that share a common property, a term that means a set of terms to which they will belong can be introduced. Terms of the ontology of higher level assign names of more common sets of concepts. For the considered example, such terms can be a term "own properties of objects" (a set of functions the range of definition of which is a set of objects of some type and the range of values depends on the term that assigns a name of a property). In this case, there also appear terms that mean auxiliary concepts. In the considered example there is an auxiliary term "types of objects" that means a set of names of types of objects (terms "chemical elements', "chemical substances", etc. can be its values).

Transition to the ontology of a higher level makes it possible to describe more common properties of a domain. Transition from the ontology of a higher level to the ontology of a lower level takes place through assigning terms owned by sets specified by terms of the ontology of a higher level.

Thus, the four-level ontology of chemistry has a structure as shown on Fig. 2. The ontology of level 4 is a chemistry metaontology. The ontology of level 3 is a set of metaontologies of sections, i.e. it is an array of the modules corresponding to the sections. Terms of the metaontology of a section are representatives of sets of terms of the metaontology of chemistry. The ontology of level 2 for each section is a set of ontologies of subsections – modules of the ontology of level 2. Each module of the ontology of level 2 contains the definition of linked sets of terms. The ontology of level 1 for each section is a set of knowledge of subsections - modules of the ontology of level 1.

For example, in the ontology of level 2 for physical chemistry [16] (Fig. 3) there are the following modules: "Properties of elements", "Properties of substances", "Properties of reactions", "Introduction to thermodynamics", "Thermodynamics. Chemical properties", "Thermodynamics. Physical properties", "Thermodynamics. Relation between physical properties and chemical properties", "Chemical kinetics". The first three modules define terms that describe properties of objects of a corresponding type. The module "Introduction to thermodynamics" defines terms used to describe general properties of thermodynamic systems and their components. Conditions of a thermodynamic system can change during a physicochemical process. Conditions of a process are assigned at discrete moments of observation. "Thermodynamics. Chemical properties" defines terms used to describe chemical changes of a substance during a process without taking into account phase changes. "Thermodynamics. Physical properties" define terms used to describe phase changes of a substance during a process without taking into account chemical changes. "Thermodynamics. Relation

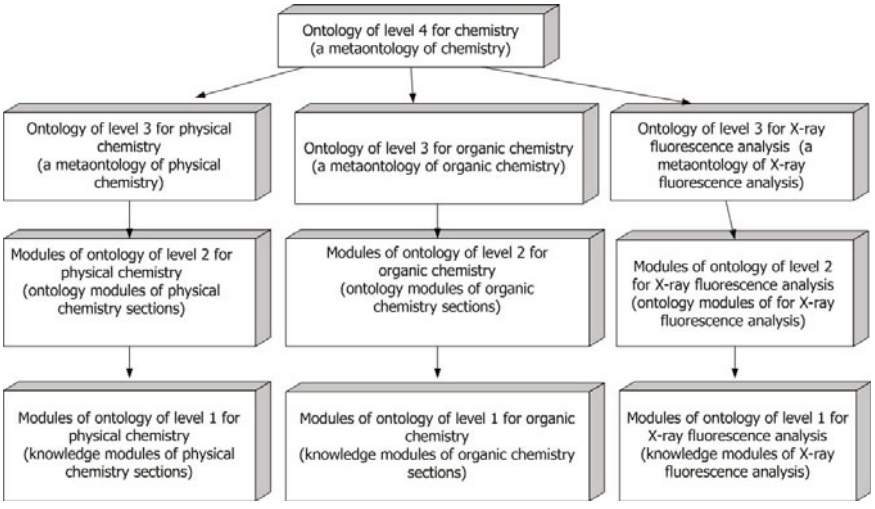


Fig. 2. The structure of the four-level ontology of chemistry

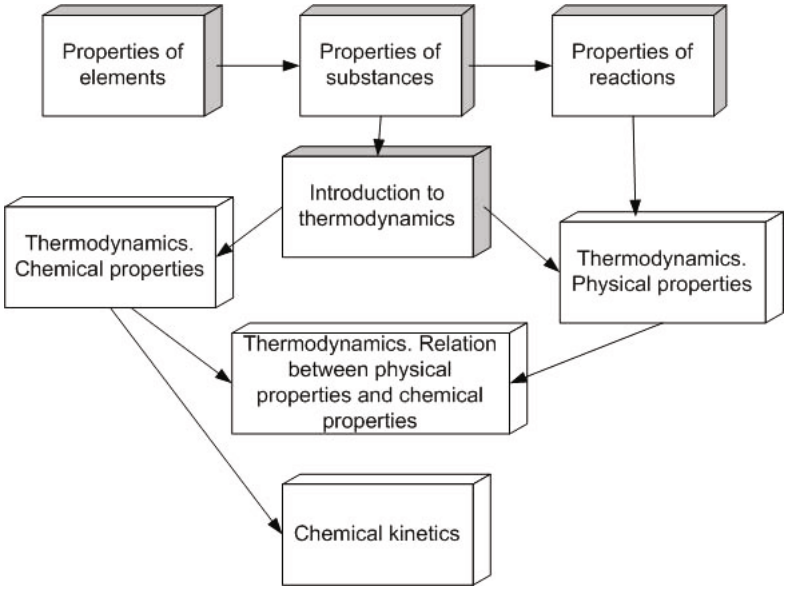


Fig. 3. The modules of the ontology of level 2 for physical chemistry

between physical properties and chemical properties" defines terms used to describe physicochemical processes. Finally, "Chemical kinetics" defines terms use to describe the dynamics of processes.

The ontology of level 2 for organic chemistry contains 26 modules [17]. It uses terms of the ontology of physical chemistry. This ontology defines terms for describing structural properties of compounds, molecular configuration, mechanisms of reactions, etc.

7 A Fragment of the Ontology of Level 4

Let us consider a fragment of the ontology of level 4 for chemistry represented by an applied logic language [24, 26] to exemplify the usage of a top-level ontology in the domain.

1. $sort\ Types\ of\ objects : \{\}N \setminus \emptyset$
   The term $Types\ of\ objects$ designates a non-empty set of names of object types for chemistry.
2. $(Type :\ Types\ of\ objects) sort\ Type : \{\}(R \cup I \cup N \cup L)$
   Each type of objects is a set of objects. Each object can be named, represented with a number, can be a logical value.
3. $sort\ Types\ of\ related\ objects : Types\ of\ objects \rightarrow \{\} Types\ of\ objects$
   The term $Types\ of\ related\ objects$ designates a function assotiating an object type with a non-empty set of names of object types.
   If $Types\ of\ related\ objects(t) = t'$ and t' is the set consisting of the elements t'_1 and t'_2 then the objects belonging to set t'_1 or to set t'_2 can be related to the objects with type t.
4. $Set\ of\ objects \equiv \{(Type : Types\ of\ objects) j(Type)\}$
   This auxiliary term designates a set of objects of all types.
5. $Own\ properties\ of\ objects \equiv (\lambda(Type : Types\ of\ objects)$
   $(\lambda(Area\ of\ possible\ values : (\{\} Value\ sets \cup \{\} Value\ corteges))$
   $(j(Type) \rightarrow Area\ of\ possible\ values))$
   The term $Own\ properties\ of\ objects$ designates a function associating an object type with a function set. The argument of each function from the set is a value set or a cortege set and the result is a function set.
   If $Own\ properties\ of\ objects(t)$ is a set consisting of function $f1$ and $f1(m)$ is a set consisting of function $f2$ then the argument of function $f2$ is an object with type t and the result is an element belonging to set m.
6. $Properties\ of\ related\ objects \equiv (\lambda(Type1 : Types\ of\ objects)$
   $(Type2 : Types\ of\ related\ objects(Type1))$
   $(\lambda(Area\ of\ possible\ values : \{\}(Value\ sets \cup \{\} Value\ corteges))$
   $(Object\ that\ has\ Type1 \rightarrow j(Type1), Object\ that\ has\ Type2 \rightarrow$
   $Related\ objects(Type1, Type2)(Object\ that\ has\ type\ 1)) \rightarrow$
   $Area\ of\ possible\ values))$
   The term $Properties\ of\ related\ objects$ designates a function associating two object types with a function set. The argument of each function from the set is a value set or a cortege set and the result is a function set. If $Properties\ of\ related\ objects(t_1, t_2)$ is a set consisting of function $f1$ and $f1(m)$ is a set consisting of function $f2$ then the arguments of function $f2$ are an object with type t_1 and its component that is an object with type t_2. The result of function f_2 is an element belonging to set m.

7. *sort Number of process steps* : $I(0, \infty)$
   The term *Number of process steps* designates a number of steps included in a physical and chemical process. This term value is a positive integer.
8. *sort Types of process objects* : $\{\}$*Types of objects* $\setminus \emptyset$
   The term *Types of process objects* designates a non-empty set of object types that are considered as components of a physical and chemical process.
9. *Process components* $\equiv (\lambda(Type : Type\ of\ process\ objects)$
   $(I[1, Number\ of\ process\ steps] \rightarrow \{\}\{(v : Set\ of\ objects) Object\ type(v) = Type\} \setminus \emptyset)$
   The term *Process components* designates a function associating an object type with a function set. If *Types of process objects*(t) is a set consisting of function $f1$ then the argument of function $f1$ is the number of a process step. The result of function $f1$ is a set of components of the process. The result is a non-empty subset of objects of type t.
10. *Properties of process components* $\equiv (\lambda(Type : Types\ of\ process\ objects)$
    $(\lambda(Area\ of\ possible\ values : \{\}(Value\ sets \cup \{\}Value\ corteges))$
    $(Step\ number \rightarrow I[1, Number\ of\ process\ steps], Process\ component \rightarrow$
    $Process\ components(Type)(Step\ number)) \rightarrow Area\ of\ possible\ values))$
    The term *Properties of process components* designates a function associating an object type with a function set. The argument of each function from the set is a value set or a cortege set and the result is a function set. If *Properties of process*(t) is a set consisting of function $f1$ and $f1(m)$ is a set consisting of function $f2$ then the arguments of function $f2$ are the index of a process step and a component of this step, that is an object of type t. The result of function $f2$ is an element belonging to set m.

8 Using the Ontology of Fourth Level

Let us now consider the example of using the ontology of level 4 when defining the ontology of level 3 for X-ray fluorescence analysis [19]. First, let us define the values of the parameters of ontology of level 4 (a set of terms of ontology of level 3).

1. *Types of objects* $\equiv \{$*Shells of chemical element atoms*,
   Radiation transition of orbital electrons, *Chemical elements*,
   Energy levels$\}$
   Object types examples for X-ray fluorescence analysis are
   Shells of chemical element atoms, *Radiation transition of orbital electrons*, *Chemical elements* and *Energy levels*.
2. *Types of related objects* $\equiv$
   $(\lambda(Type : \{$*Shells of chemical element atoms*,
   Radiation transition of orbital electrons, *Chemical elements*,
   Energy levels$\})$
   $(If\ Type\ = Chemical\ elements$
   then $\{$*Radiation transition of orbital electrons*, *Energy levels*$\})$,
   $(If\ Type \neq Chemical\ elements\ then\ \emptyset\}$

Objects of *Chemical elements* type are exclusively related to objects of *Radiation transition of orbital electrons* and *Energy levels* types while objects of other types are not related. To define the result of function *Types of related objects* for any object types, conditional expressions are used.

3. *Types of process objects* $\equiv \{$*Chemical elements*, *Radiant energies*$\}$
   Chemical elements and radiant energies are considered for every physical and chemical processes.

Now let us define examples of ontological constraints that are part of ontology of level 3.

1. *Shells of chemical element atoms* $\subset \{\}N \setminus \emptyset$
   The term *Shells of chemical element atoms* designates a non-empty set of names.
2. *Energy levels* $\subset \{\}N \setminus \emptyset$
   The term *Energy levels* designates a non-empty set of names.
3. *Chemical elements* $\subset \{\}N \setminus \emptyset$
   The term *Chemical elements* designates a non-empty set of names.
4. *Radiation transition of orbital electrons* $\subset \{\}N \setminus \emptyset$
   The term *Radiation transition of orbital electrons* is a non-empty set of names.

Then let us define examples of terms that are part of ontology of level 3 and are names of functions.

1. *Own properties of shells* $\equiv$
   Own properties of objects(*Shells of chemical element atoms*)
   The term *Own properties of shells* designates a function set which is the result of function *Own properties of objects* applied to argument *Shells of chemical element atoms*.
2. *Own properties of radiation transitions* $\equiv$
   Own properties of objects(*Radiation transition of orbital electrons*)
   The term *Own properties of radiation transitions* designates a function set which is the result of function *Own properties of objects* applied to argument *Radiation transition of orbital electrons*.
3. *Properties of radiation transition of orbital electrons of elements* $\equiv$
   Properties of related objects(*Chemical elements*,
   Radiation transition of orbital electrons)
   The term *Properties of radiation transition of orbital electrons of elements* designates a function set which is the result of function *Properties of related objects* applied to arguments *Chemical elements* and *Radiation transition of orbital electron*.
4. *Properties of energy level for an element* $\equiv$
   Properties of related objects(*Chemical elements*,
   Energy levels)
   The term *Properties of energy level for an element* designates a function set which is the result of function *Properties of related objects* applied to arguments *Chemical elements* and *Energy levels*.

5. *Properties of elements of a sample* ≡
   Properties of process components(*Chemical elements*)
   The term *Properties of elements of a sample* designates a function set which is the result of function *Properties of process components* applied to argument *Chemical elements*.

Finally, let us define examples of terms that are part of ontology of level 2.

1. *sort Binding energy of electrons on an energy level for an element* :
   Properties of energy levels for an element($R(0, \infty)$)
   The term *Binding energy of electrons on an energy level for an element* designates a function being an element of the function set defined through *Properties of energy levels for an element*($R(0, \infty)$). The first argument of the function belongs to object set *Chemical elements* and the second one belongs to object set *Energy levels*. The result of the function is a positive real number.
2. *sort Characteristic radiation frequency* : *Properties of radiation transition of orbital electrons of elements*($R(0, \infty)$)
   The term *Characteristic radiation frequency* designates a function whose the first argument belongs to object set *Chemical elements* and the second one belongs to object set *Radiation transition of orbital electrons*. The result of the function is a positive real number.
3. *sort* $Wave - length$ *of characteristic radiation* : *Properties of radiation transition of orbital electrons of elements*($R(0, \infty)$)
   The term $Wave - length$ *of characteristic radiation* designates a function with its first argument belonging to object set *Chemical elements* and second belonging to object set *Radiation transition of orbital electrons*. The result of the function is a positive real number.
4. *sort Energy of characteristic radiation* : *Properties of radiation transition of orbital electrons of elements*($R(0, \infty)$)
   The term *Energy of characteristic radiation* designates a function whose the first argument of the function belongs to object set *Chemical elements* and the second one belongs to object set *Radiation transition of orbital electrons*. The result of the function is a positive real number.

9 Distinguishing Features of This Ontology Development Method

Let us consider distinguishing features of the proposed ontology development method for domains with complicated structures.

All other ontology development methodologies propose that development should begin from the analysis of the scope and purpose of an ontology. If it is used to work with domain knowledge, the verbalized information is this knowledge. Then it can be assumed that the ontology of level 2 contains

nothing but terms of the ontology of knowledge. In this case, the tasks to be solved are tasks of a search for information in the knowledge, task of filling in databases, also with the help of data mining methods. The ontology of level 3 describes structures of different knowledge ontologies (different sections of one domain or different domains). The ontologies of all the following levels describe structures of knowledge metaontologies.

If an ontology is used to create databases containing domain information, the ontology of level 2 contains nothing but terms of the ontology of data. The ontology of level 3 describes structures of different data ontologies (different sections of one domain or different domains). The ontologies of all the following levels describe structures of data metaontologies.

If an ontology is used to create a knowledge-based system to solve applied tasks different from problems of structuring, editing and searching for information, the ontology of level 2 contains terms of the two ontologies and ontological constraints include a group of constraints that assign the interrelations between the knowledge and the reality. The division of the ontology into two parts remains on all the following levels.

Reusing data, knowledge and programs is one of the purposes of ontologies. A knowledge-based system can be reused if it is known what applied tasks are to be solved. Achieving this purpose requires the development of an ontology of tasks that determines input and output parameters of a task, relations between these parameters and terms of a domain ontology.

If only an ontology of level 2 is developed, an ontology of tasks contains specified terms of a domain for assigning names of input and output parameters of a task.

Transition to the metaontology (ontology of level 3 and higher) inputs names of sets of terms. Then an ontology of tasks uses names of sets to assign input and output parameters of a task. Consequently, problem-solving methods developed in terms of metaontologies will be more general as compared to those developed in terms of the ontology of level 2.

A class to which a given object can belong to is one of characteristics of domain objects. For example, in the domain of chemistry there are different classes of chemical elements (e.g., metals, nonmetals, etc.), chemical substances, reactions, etc. Besides, in this domain there are different classifying principles for chemical substances (e.g., classes according to functional group, classes according to a type of carbon skeleton). There are also hierarchies of classes and each method of classification can have its own hierarchies. The information about relations between classes is represented on level 2 in the ontology of knowledge. Level 3 can have a term with the meaning “classifying principle for chemical substances”. To solve a task such property as class can also be used. But it, for example, already characterizes an element of a set of substances that is a reactant and is described in the ontology by a term with the stated meaning. Description of a hierarchy of classes in accordance with one classifying principle in the ontology narrows the ontology application area.

10 Intelligent Systems for Domains with Complicated Structures

An intelligent system for a domain with complicated structures [27] must be able to store the ontology and knowledge of each section and subsection and, when solving applied tasks, to ensure the usage of those informational components that are necessary in this case, i.e. to permit the integration of knowledge and ontologies of different levels within the framework of one information resource. Such integration can be provided by the ontology with several level.

Editors of ontologies of different levels and knowledge editors ensure the development of the information content; creation and editing of a module of the ontology of level i are managed by the ontology of level i+1, knowledge editing is managed by the correspondent ontology of level 1.

Different nonstandard quantities whose structures are described in terms of their own specific ontologies are a distinctive feature of ontologies considered in the paper. Such quantity for organic chemistry can be exemplified by a structural formula. If it is required to input information about the value of nonstandard quantity, one can use a special editor that "knows" the structure of representation of the value of this quantity and uses the value assignment way accepted in the domain (e.g., for a structural formula one can use graphic editor). A special editor call is controlled by the ontology of level 1. Elements of nonstandard quantity can be used to generate knowledge base and assign source data of tasks solved by the system.

The development of the domain leads to changes in the composition of nonstandard quantities that are supported by the subsystem of work with such quantities (Fig. 4) that allows to assign the name of a new quantity and associate the special editor.

The consequence of changing ontologies of different levels is changes in a set of classes of tasks solved by the intelligent system which requires mechanisms of development of not only information components but also program ones of the system, i.e. subsystems for solving tasks.

RDBMS stores modules of ontologies of all levels and modules of knowledge. Each term is associated with its table. The knowledge base structure is correspondent to the ontology structure. Each term described in the ontology is associated with a table in the knowledge base. The knowledge base scheme to represent knowledge is automatically defined by assigning terms and connections between them in the ontology.

An ontology used to create intelligent systems intended not only for storage of, search for and editing of ontologies and knowledge but also for solving other tasks of a domain is supposed to define a system of concepts which assigns source data of such tasks and represents solutions. The system of concepts for chemistry defines different properties of physicochemical processes for different steps of those processes.

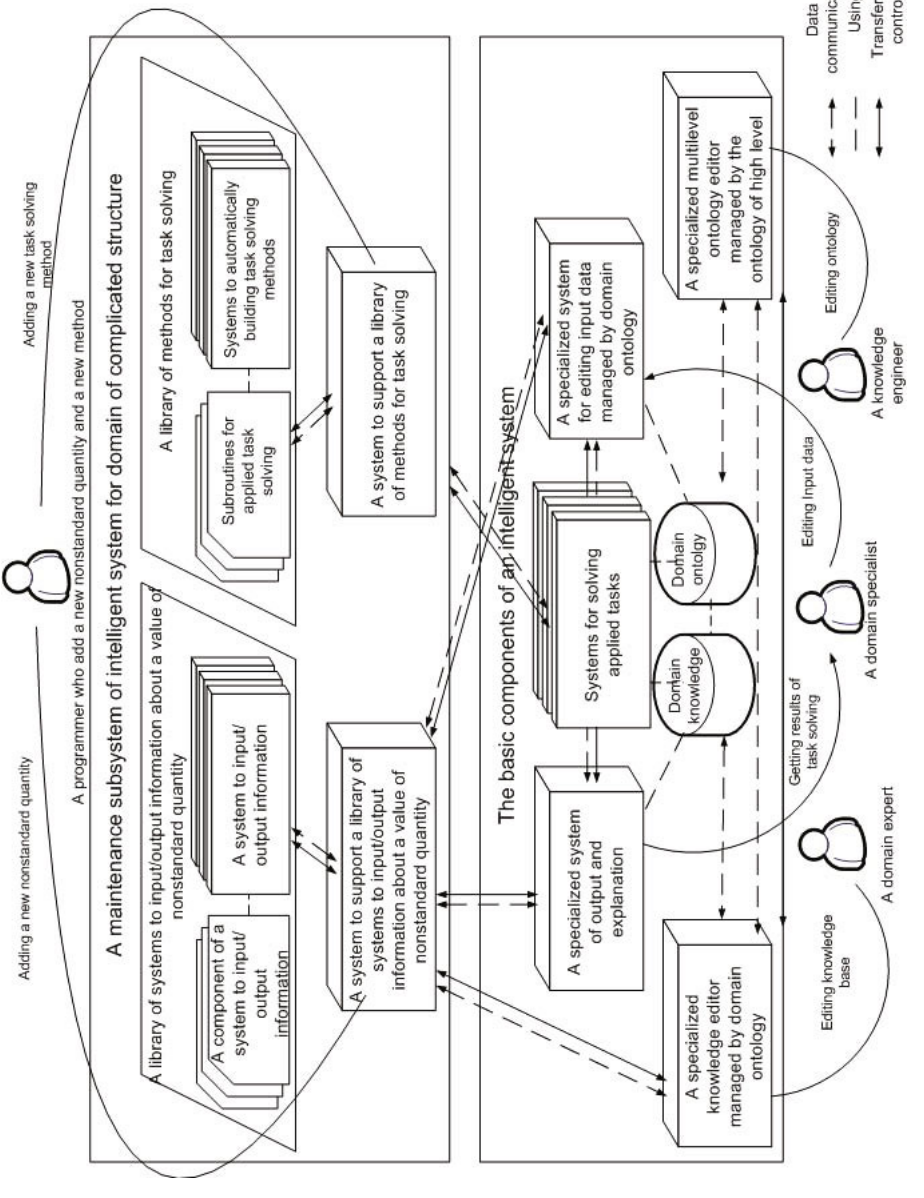


Fig. 4. The components of an extendable intelligent system for a domain with complicated structure

Ontology model terms are used to describe problem-solving methods. The higher an ontology level is, the more general is the method. The method description is used to develop a system for solving tasks of a correspondent class – the program component of the intelligent system for chemistry.

Although all main classes of applications are defined during the analysis of a domain and development of its multilevel ontology, further development of the ontology can lead to the increase in classes of tasks or more effective problem-solving methods that are particular cases of the existing tasks. This requires a subsystem of support of library of problem-solving methods that can allow to add another subsystem and ensure its usage by specialists.

Thus, users of an intelligent system for a domain with a complicated structure include:

- programmers adding subsystems to work with nonstandard quantities and subprograms to perform new classes of applications;
- knowledge engineers adding ontologies of different levels using a special editor of multilevel ontology controlled by an ontology of a domain with complicated structure;
- experts adding knowledge of new sections of a domain;
- specialists using information and program components to perform applications.

So, a programmer is the user of a maintenance system, a knowledge engineer is the user of an ontology editor, a domain expert is the user of knowledge editor. They could create the components of an intelligent system used by a domain specialist to solve applied tasks.

11 Conclusion

The described method for developing multilevel ontologies was tested by creating the multilevel ontology of chemistry and its sections [16,19] and of optimizing programs [20]. The developed ontologies were used to create knowledge-based systems for sections of chemistry. At present, the multilevel model of chemistry is used to create the web-based intelligent system [21] for chemistry whose information content will include ontologies and knowledge of these sections and the software content will allow chemists to solve the tasks of these sections. This method has been tested when teaching knowledge-based system development methods to the students of the Far Eastern National University about the ten years.

Acknowledgments. This paper was made according to the program "The fundamental problems of computer sciences and information technologies" of fundamental scientific research of the Presidium of the Russian Academy of Sciences, the project "Intellectual systems based on multilevel models of domains".

References

1. Kleshchev, A.S.: Using Ontologies for Software Development. In: Russian Conf. Knowledge-Ontologies-Theories, vol. 1, pp. 122–129. Institute of Mathematics, Novosibirsk (2007) (in Russian)
2. Jasper, R., Uschold, M.: A Framework for Understanding and Classifying Ontology Applications, http://www.informatik.uni-trier.de/ ley/db/indices/a-tree/u/Uschold:Michael.html
3. Staab, S., Maedche, A.: Knowledge Portals: Ontologies at Work. AI Magazine 22(2), 63–75 (2001)
4. Zhdanova, A.V.: The People's Portal: Ontology Management on Community Portals, http://www.ee.surrey.ac.uk/Personal/A.Zhdanova/publications.htm
5. Kleshchev, A.S., Gribova, V.V.: From an Ontology-oriented Approach to User Interface Development. Int J. Information Theory & Applications 10(1), 87–93 (2003)
6. Denny, M.: Ontology Building: a Survey of Editing Tools, http://www.xml.com/pub/a/2004/07/14/onto.html
7. Kleshchev, A.S., Orlov, V.A.: Computer Knowledge Bank. A Universal Direction in Solving the Problem of Editing Information. Information Technologies. 5, 25–31 (2006) (in Russian)
8. http://www.alphaworks.ibm.com/contentnr/semanticsfaqs
9. http://www.w3.org/TR/webont-req/
10. Corcho, O., Fernandez-Lopez, M., Gomez-Perez, A.: Methodologies, Tools and Languages for Building Ontologies. Where is their Meeting Point? Data & Knowledge Engineering 46, 41–64 (2003)
11. Cristani, M., Cuel, R.: A Survey on Ontology Creation Methodologies. Int. J. on Semantic Web & Information Systems 1(2), 48–68 (2005)
12. Jones, D., Bench-Capon, T., Visser, P.: Methodologies for Ontology Development, http://www.iet.com/Projects/RKF/SME/methodologies-for-ontology-development.pdf
13. Noy, N., McGuinness, D.L.: Ontology Development 101: A Guide to Creating Your First Ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001- 0880 (March 2001), http://protege.stanford.edu/publications/ontology_development/ontology101.html
14. Guarino, N., Carrata, M., Giaretta, P.: An Ontology of Meta-level Categories. In: 4th Int. Conf. Principles of Knowledge Representation and Reasoning, pp. 270–280. Morgan Kaufman, San Mateo (1994)
15. Kleshchev, A.S., Artemjeva, I.L.: An analysis of some relations among domain ontologies. Int. Journal on Inf. Theories and Appl. 12(1), 85–93 (2005)
16. Artemieva, I.L., Tsvetnikov, V.A.: A Fragment of the Ontology of Physical Chemistry and its Model. Investigated in Russia [Electronic resource]: Multysubject Scientific J. 5, 454–474 (2002) (in Russian), http://zhurnal.ape.relarn.ru/articles/2002/042.pdf
17. Artemieva, I.L., Vysotsky, V.I., Reshtanenko, N.V.: A Model for the Ontology of Organic Chemistry. Scientific & Technical Information 8, 19–27 (2005) (in Russian)
18. Artemieva, I.L., Vysotsky, V.I., Reshtanenko, N.V.: Description of Structural Formula of Organic Compounds in the Model for the Ontology of Organic Chemistry. Scientific & Technical Information 2, 11–19 (2006) (in Russian)

19. Artemieva, I.L., Miroshnichenko, N.L.: A Model for the Ontology of X-ray Fluorescence Analysis. Informatic & Management Systems 2, 78–88 (2005) (in Russian)
20. Artemjeva, I.L., Knyazeva, M.A., Kupnevich, O.A.: Processing of Knowledge about Optimization of Classical Optimizing Transformations. Int. J. Information Theories and Applications. 10(2), 126–131 (2003)
21. Artemieva, I.L., Reshtanenko, N.V.: Specialized Computer Knowledge Bank for Organic Chemistry and its Development Based on the Ontology. Artificial Intelligence, Ukraine 4, 95–106 (2006) (in Russian)
22. Artemieva, I.L.: Multilevel Ontologies for Domains with Complicated Structures. In: XIII-th Int. Conf. Knowledge-Dialog-Solution, vol. 2, pp. 403–410. FOI ITHEA, Bulgaria (2007)
23. Kleshchev, A.S., Artemjeva, I.L.: Mathematical Models of Domain Ontologies. Int. J. Inf. Theories and Appl. 14(1), 35–43 (2007)
24. Kleshchev, A.S., Artemjeva, I.L.: A Mathematical Apparatus for Domain Ontology Simulation. An Extendable Language of Applied Logic. Int. J. Inf. Theories and Appl. 12(2), 149–157 (2005)
25. Kleshchev, A.S., Artemjeva, I.L.: A Mathematical Apparatus for Ontology Simulation. Specialized Extensions of the Extendable Language of Applied Logic. Int. J. Inf. Theories and Appl. 12(3), 265–271 (2005)
26. Kleshchev, A.S., Artemjeva, I.L.: A Mathematical Apparatus for Domain Ontology Simulation. Logical Relationship Systems. Int. J. Inf. Theories and Appl. 12(4), 343–351 (2005)
27. Artemieva, I.L.: Intellectual Systems for Domains with Complicated Structures. Scientific and Technical Journal of the Saint-Petersburg Technical University 2, 5–15 (2008) (in Russian)

Technology of Ontology Building for Knowledge Portals on Humanities*

Yury Zagorulko and Olesya Borovikova

A.P. Ershov Institute of Informatics Systems
Siberian Branch of the Russian Academy of Sciences
6, Acad. Lavrentjev ave., 630090, Novosibirsk, Russia
{zagor,olesya}@iis.nsk.su

Abstract. The paper presents a technology of ontology building for specialized Internet portals providing content-based access to scientific knowledge and information resources related to humanities. The information basis for such portals is formed by ontologies, which allows heterogeneous data and knowledge to be presented in a unified manner ensuring their relatedness. Based on the ontology, internal storages of the portal data are constructed and management of its information content, as well as navigation and search, are organized. To meet the portal objectives, the ontology should be well-structured and adequately present its problem and subject domains. Therefore the portal ontology is divided into the domain-independent ontologies and subject domain ontology. The technology of ontology building includes the methods of ontology building, the ontology description language and ontology editor. The methods of ontology building are defined by its structure and supported by the facilities of the ontology editor. The ontology description language and ontology editor are selected and designed in such a way that they are easy to understand and use for experts in humanities. The ontology editor is also designed taking into account its use in the distributed development of ontologies.

Keywords: Knowledge portal, ontology, content-based access, technology of ontology building, ontology editor.

1 Introduction

Recently, a great amount of scientific knowledge and information resources relating to humanities has been accumulated in the Internet. However, the access to and the use of these knowledge and resources is rather complicated as they are disembodied and ill-structured, or distributed over various Internet sites, electronic libraries and archives. At the same time, researchers need an efficient access to scientific papers and other information resources containing descriptions of methods and approaches developed in the framework of the field of science interesting to them.

* The authors are grateful to the Russian Foundation for the Humanities (grant 07-04-12149) for financial support of this work.

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 203–216, 2011.

To meet the need described above, we have suggested a conception and architecture of specialized Internet portals – knowledge portals [1] intended to provide systematization and integration of knowledge and information resources related to a given field of science, as well as content-based access to them from any Internet spot. In addition, according to the conception, the knowledge portal should not only provide the access to its own information resources, but also support effective navigation through relevant Internet resources previously remarked (indexed).

In addition to support flexible and consistent representation of some field of science (using special entities like persons, organizations, events, methods, objects and results of researches etc.), as well as content-based access to integrated knowledge and information resources, an important requirement to the knowledge portal is the possibility of its declarative adjustability to any area of knowledge both at the development and operation stage. This possibility allows us to track up the dynamics of appearance of new knowledge and information resources related to the topics of portal and in that way to secure maintenance of its topicality and utility.

The requirements described above were satisfied owing to the fact that ontology was chosen as a conceptual basis and an information model of the knowledge portal. Thus, the problem of ontology development for the scientific knowledge portal is rather actual. The paper describes our experience in designing and using such an ontology.

2 Requirements to Knowledge Portal Ontology

Above all we have to clarify what we mean by "ontology". We use the concept of "ontology" in the sense as it is used in computer science and artificial intelligence. Basing on the definitions presented in [2–5], we can say that an ontology is an explicit specification (model) of some part of the world as applied to a specific area of interests. In the context of this paper, an ontology will present a description of a certain branch of humanities and research activity related to it.

Let us consider the main requirements to the portal ontology.

The ontology should not only provide a formal presentation of concepts of the subject domain of the portal, but also support all necessary functionality, i.e. it should afford a basis for effective representation of diverse information related to the portal topic and provide a convenient content-based access to it. Besides, the ontology should provide integration of relevant information resources into information space of the knowledge portal and convenient navigation through it.

To support an effective representation of the subject domain knowledge, the ontology should provide a description of concepts with a complex structure and diverse semantic relations between them. An important requirement to the portal ontology is the possibility to order subject domain concepts in a "generic-specific" hierarchy and to support inheritance of properties through this hierarchy.

Since the ontology should provide the content-based declarative adjustment of the portal on a given area of knowledge and support its functionality, it should

be designed in such a way that it could be used for automatic generation of the following components: the portal data base scheme (the logical structure of DB and data integrity constraints), forms for data base filling (with information objects being instances of the ontology concept), the scheme of navigation through the portal information space (along with the ontology relations), and query forms (using the ontology relations and concepts).

To make the knowledge portal easily adjustable to any area of knowledge, we should define the ontology structures that are independent of the subject domain of the portal.

Moreover, the ontology should have certain properties, such as extensibility and integrability with an existing ontology.

3 Structure of Knowledge Portal Ontology

3.1 Definition of Portal Ontology

An ontology satisfying the requirements described above can be represented by following structure:

$$O = \langle C, A, R_C, T, D, R_A, F \rangle, \tag{1}$$

where

C is a set of classes describing the concepts of a certain problem or subject domain;

A is a set of attributes describing the properties of concepts and relations;

$R_C = \{r_C | r_C \subseteq C \times C\}$ is a set of relations defined on classes (concepts);

T is a set of standard types of attribute values (*string, integer, real, date*);

D is a set of domains (the sets of values of a standard type *string*);

$R_A = R_{AT} \cup R_{AD}$, where $R_{AT} \subseteq A \times T$ is a relation that links attributes with the data types of their admissible values, $R_{AD} \subseteq A \times D$ is a relation that defines a discrete set of values (a domain) for each attribute;

F is a set of constraints on the values of attributes of concepts and relations.

From the informal point of view the portal ontology serves for representing concepts that are required for the description of both research activity and scientific knowledge in whole and specific areas of knowledge in particular.

3.2 Structuring of Knowledge Portal Ontology

To meet the portal objectives, the ontology should be well-structured and adequately present its problem and subject domains. Therefore the portal ontology is divided into domain-independent (basic) ontologies and the subject domain ontology.

The basic ontologies are the ontology of research activity and the ontology of scientific knowledge (see Figure 1) that are independent from the subject domain of the portal.

The ontology of research activity is based on the ontology suggested in [6]. Practically, it is a top-level ontology that includes the basic classes of concepts

related to the research activity management such as *Person* (*Researcher*), *Organization*, *Event*, *Activity*, *Publication*. These classes are used when describing the participants of research activity, scientific events, research programs and projects, various types of publications and the materials represented in a printed or electronic format (such as monographs, articles, reports, proceedings of conferences, periodicals, photo and video data, etc.). This ontology includes a class *Information resource* that serves for description of relevant information resources presented in the Internet.

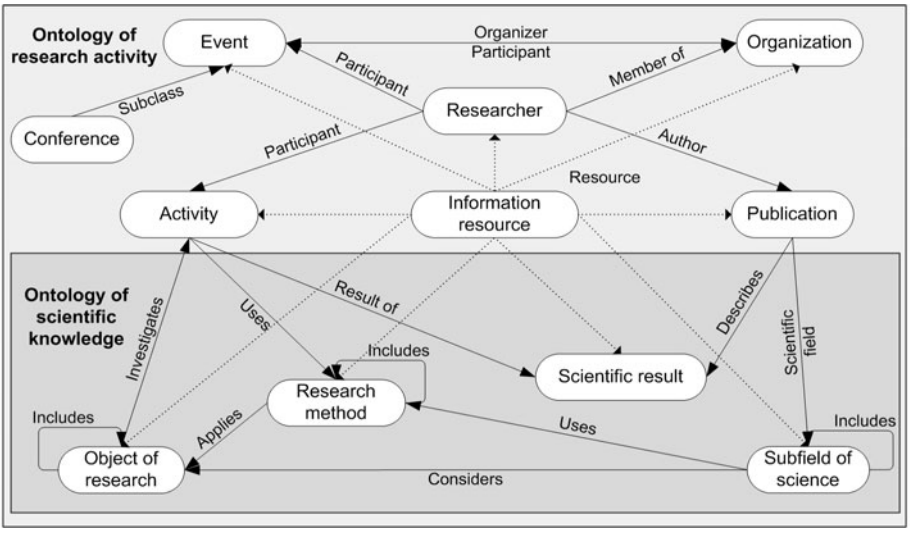


Fig. 1. The basic ontologies of a knowledge portal

The ontology of scientific knowledge is virtually a meta-ontology. It states the main structures that are used for building the lower-level ontologies, i.e. the subject domain ontologies describing some specific areas of knowledge or fields of science. In particular, this ontology contains the meta-concepts that specify the structures for the description of concepts of a specific subject domain, such as *Subfield of science*, *Research method*, *Subject or Object of research*, *Scientific result*. Using these meta-concepts, we can describe fields and subfields significant for a given science, determine a classification of methods and objects of research, and describe the results of research activity.

The concepts of the ontology of scientific knowledge are interconnected with each other and with the concepts of the ontology of research activity by associative relations, the main of which are the following:

- "Scientific field" – links events, publications, organizations, persons or information resources with subfields of science;
- "Describes" – links a publication with a scientific result, method or object of research;

– "Uses" – links a research method with activity, person or organization;
– "Investigates" – attaches some activity to the object of research;
– "Result of" – serves for linking scientific results with research activity;
– "Resource" – links information resources with any concept of the ontology;
– "Member of" – links a person with the organization where he works.

Note that the last relation has three additional attributes "appointment", "date of hiring" and "date of dismissal" that allow us to specify the position and working period for a person.

These associative relations have been chosen taking into account not only completeness of presentation of the problem and subject domain of the portal, but also convenience of navigation through the portal information space and of content-based search.

The subject domain ontology of the knowledge portal describes some field of science. It is built for organizing an effective access to information resources related to a certain research area, so it should meet the requirements described in section 1.

The concepts of the subject domain ontology are at the same time realizations of meta-concepts of the ontology of scientific knowledge and can be ordered in the "generic-specific" hierarchy.

As a rule the ontology of a concrete subject domain includes four basic hierarchies: *Subfield of science*, *Research method*, *Object of research*, *Scientific result*.

4 Technology of Ontology Building

The technology of ontology building includes the ontology description language, ontology editor that supports ontology constructing, and methods of ontology building.

The ontology description language and ontology editor were selected and designed in such a way that they are easy to understand and use for experts in humanities. In particular, to conform to these requirements, we refused to use such popular means as the ontology representation language OWL [7] and editor Protege [8] (at the same time we provided the possibility of translation the developed ontology to OWL representation).

Besides, the ontology editor was also designed taking into account its use in the distributed development of ontologies.

4.1 Ontology Description Language

The knowledge representation language of Semp-TAO system [9], a tried-and-true one, was taken as a basis of the ontology description language.

The classes of concepts in this language are described as follows:

```
class Name_of_Class (Class_Parent);
        Description_of_Attributtes;
constraints
        Description_of_Constraints;
end;
```

Below we give an example of simplified descriptions of the class *Person* and its successor – the class *Reseacher*:

```
class Person;
        Family_Name: string;
        Name: string;
        Patronymic_Name: string;
        Sex: Sex;
        Date_of_birth: date;
        Date_of_death: date;
constraints
        Date_of_birth < Date_of_death;
end;

class Reseacher (Person);
        Academic_Degree: Academic_Degree;
        Academic_Sstatus: Academic_Sstatus;
        E-mail: string;
        Office_Phone: string;
end;
```

The description of a relation looks like:

```
relation Name_of_Relation (Name_of_Argument1: Class1;
                           Name_of_Argument2: Class2);
        Mathematical_Properties_of_Relation;
        Description_of_Attributtes;
constraints
        Description_of_Constraints;
end;
```

Relations can have mathematical properties, such as transitivity, symmetry or reflexivity.

Let us give an example of the relation "Work in"

```
relation Work_in (who: Person; where: Organization);
        Appointment: Appointment;
        Date_of hiring: date;
        Date_of_dismissal: date;
constraints
        Date_of_hiring > Date_of_Birth + 18;
        Date_of_dismissal > Date_of_hiring;
end;
```

Domains are described in the following way:

```
Domain Name_of_Domain = Set_of_String_Values;
```

Let us give examples of descriptions of some domains:

```
Domain Sex = {mail, female};
Domain Position = {Director, Head_of_Laboratory, Researcher,
        Senior_Researcher, Junior_Researcher, Laboratory_Assistant};
```

4.2 Ontology Editor

The ontology editor is intended for ontology building by means of the language described in section 4.1. It includes graphical interface that simplifies the development of ontology and facilities that ensure its correctness.

The ontology editor is implemented as a Web-application accessible to authorized users. To make possible a distributed development of ontologies, the ontology editor has a procedure for granting privileges to experts of different levels.

Using the ontology editor, an expert can create, modify and delete any elements of the ontology: classes of concepts, relations, and domains.

When a class is created, it takes its name, a set of attributes that define various properties of concepts, as well as constraints on the attributes values. A parent of the class under creation can be selected from the set of already created classes. Thereby this class inherits from the parent class not only all its attributes, but also its relations, whereas the parent class gets linked to a new class by "subclass" ("is-a") relation.

For each attribute of the class, its name and status (mandatory or not), the range of values (type or domain), and the number of possible values (one or a set) are defined.

The domain is described by its name and the set of elementary (string) values. For each value from the domain, an expert can define the language (Russian or English).

The ontology classes can be linked by directed binary relations. The peculiarity of these relations is their (relations') ability to have their own attributes that specify the nature of the link between the relation's arguments.

To make the presentation of information more convenient for a user of the portal, the possibility of adjustment of knowledge and data visualization is provided. For this purpose, the templates of visualization are created for objects of each class of the ontology and for references to these objects.

A template of visualization for a class of objects contains all its attributes and all relations associated with it. By default, its attributes and relations are depicted in the same order as defined in the ontology, but the user can change this order.

A template of visualization for a reference to a class object can include both the attributes of this class and the attributes of classes that are linked with it by relations and attributes of these relations. The values of attributes included in this template are used for building a text representation of a hyperlink to a class object.

4.3 Features of Methodology of Ontology Building for a Knowledge Portal

The methodology of ontology building for a knowledge portal has much in common with other well-known methodologies that use core ontologies, for example Cyc [10] or SENSUS methodologies [11], but it has its own features. On the one hand, this methodology is wholly defined by the ontology structure defined in part 3, and on the other hand it is supported and at the same time restricted by the ontology editor facilities.

The subject domain ontology building is founded on usage of the basic classes (meta-concepts), defined in the ontology of scientific knowledge, as the root classes of such hierarchies as the hierarchy of subfields of science, hierarchy of research methods, hierarchy of objects of research, and hierarchy of scientific results.

When building a subject domain, new classes are defined as descendants of the basic classes. These classes are automatically inserted into the corresponding hierarchies by means of the inheritance relation. Let us remind that inheritance is implemented in such a way that the derived class inherits all attributes and relations from the parent class. Besides, this new class can have its own properties represented, as usual, by means of attributes and constraints and by binary relations linking this class with other classes.

Note that the specific character of the subject domain may need introduction of new classes which will not be descendants of the five basic classes described above. Extension of the ontology of research activity may also be required. This can be done by either introduction of new classes that are descendants of the basic classes of this ontology or definition of new classes independent of the basic ones.

Formal descriptions of concepts and relations so defined determine structures for representing real objects existing in some subject domain and provide their interconnections.

When the knowledge portal is already in service, new knowledge on its subject domain can be found or gaps and inaccuracies in the presented knowledge can be revealed. Undoubtedly this situation requires evolving and correcting the ontology. However, when the ontology is edited, it is necessary to maintain consistency of the portal knowledge system and exclude loss of information [12].

Modification of the ontology can consist in extension or rebuilding of its system of concepts, as well as addition, deletion or renaming of concepts, relations and/or attributes.

First, let us consider the cases concerned with extension of the system of concepts (conceptual framework).

In the simplest case, this extension consists in addition of a new attribute to some concept. Here we must take into account that the same attribute can belong to one of the concepts being a descendant of the editing concept. Therefore we have to look through all such descendants and if necessary to rename the corresponding attributes.

Addition of a new concept to a lower level of the concept hierarchy does not take any efforts for maintaining consistency of the portal's knowledge system, as in this case the new concept will simply inherit all attributes and relations of the higher level concepts.

When adding a concept which will become a root concept for one of the concept hierarchies, it is necessary to consider the attributes and relations of all concepts in this hierarchy. It is possible that we'll need to remove a part of attributes and relations from these concepts to the new concept. We should perform this action also taking into account the possibility of appearance (creation) of new branches of the hierarchy springing from the new concept.

Insertion of a new concept into the hierarchy between two "old" concepts also requires some methodological efforts. To avoid duplication and possible conflicts of names, it is necessary to select carefully the attributes and relations for the new concept from the lower level concepts.

When deleting a "leaf" concept, i.e. a concept situated at the lowest level of the hierarchy, to avoid loss of knowledge and data, it is necessary to consider the possibility of transferring its attributes and relations to some concept from the higher level of the hierarchy. We should realize that if there exist information objects created on the basis of the deleting concept then, to avoid loss of data, it is necessary to "attach" them to the parent of the deleting concept. But it can appear that such "attachment" is not enough for maintenance of all information on these objects if the attributes and relations of the deleting concept were not previously transferred to the higher level (parent) concept.

If the deleting concept is not a "leaf" concept, before its deletion we should consider the possibility of transferring its attributes and relations to one of its descendants. Analogously to the case with a "leaf" concept, the corresponding information objects should be "attached" to the parent concept and modified in accordance with its structure.

Deletion of "root" concepts of the knowledge portal ontology being in service or at the stage of its content creation is not recommended because of possible loss of information.

When deleting attributes from concepts, we also should take into account possible loss of information. A particular case of deletion of an attribute is its moving to the higher level or lower level concept. As a result of this moving, the attribute can become more general or more specific. In the former case information is not lost because in any case the moving attribute will be inherited by the modifying concept. In the latter case loss of information is possible, therefore we have to take measures for information recovery.

Sometimes moving of a concept within a hierarchy is required. In this case we should take into account that the sets of attributes and relations inherited by a concept are changed. Possibly, we'll have to manually restore some attributes and relations lost after this moving.

Moving subtrees from one branch of the hierarchy to another is a rather interesting case of evolution of the ontology. Virtually, this case is recursively

reduced to the aforesaid. For the most part it is enough to "put in order" the root concept of the moving subtree, and the rest concepts of this subtree will be corrected automatically.

5 Content-Based Access to Portal Content

By introduction of formal descriptions of the subject domain concepts in the form of classes of objects and relations between them, the portal ontology defines structures for presentation of real data (objects) and relations between them. The portal data themselves are presented as a set of linked information objects.

Each information object (IO) corresponds to a certain class of ontology (it is an instance of this class) and presents a description of a certain object of the subject domain. There may be connections between information objects whose semantics is defined by relations between the corresponding classes of ontology.

Description of information resources is an important component of the information content of a portal. According to the definition of the ontology of research activity from section 3.2, each resource corresponds to such a concept of the ontology as Information resource, and its description includes an instance of this concept and a set of instances of relations that links it with the instances of other concepts of the ontology. The set of attributes and relations of Information resource is based on Dublin Core standard [13] and includes the following units: Title of the resource, Address in the Internet (URL), Subject of the resource, Resource type, Language, etc. Each resource can be linked by relations with persons, organizations, events, subfields of sciences etc.

Information content of a portal is produced (formed) by the expert with the help of a data editor that allows one to create, modify and delete information objects and relations between them. Operation of the data editor is based on the portal ontology. When a new information object is created, first of all, the expert selects the corresponding class of the ontology. Then, based on descriptions of this class and its relations, an input form for the corresponding information object is automatically generated. This form includes fields for input of the values of the object attributes and its relations with other objects already existing in the content of the portal.

Thus, the information content of the portal includes both general knowledge (presented in the ontology) and knowledge on concrete objects of the subject domain and their connections (presented by information objects and relations between them).

Navigation through the portal information space is realized in accordance with the content of its ontology. A navigation engine provides transition from the concepts of the ontology to their instances (lists of information objects) and then transition along ontological links (relations) from one information object to another.

Since the search is also based on the ontology, the user can formulate his query in terms of the portal subject domain. The basic elements of the query are concepts and relations of the ontology, as well as constraints on the required data.

Retrieval queries are formed by means of a special graphic interface driven by the portal ontology. When a user selects a class of the sought-for information objects, a retrieval form is generated where the user can define constraints on values of both the attributes of the sought-for object and objects connected with it by associative relations.

For example, the query "Find Gumilev's publications on the ethnic history in the period from 1962 to 1989" formally looks as follows:

```
Class "Publication"
      Attribute "Date of publication": (>=1962)&(<=1989)
      Relation "Author":
          Class "Researcher"
                Attribute "Name" = "Gumilev"
      Relation "Scientific field":
          Class "Subfield of science"
                Attribute "Name of subfield" = "Ethnic history".
```

6 Building an Ontology for Knowledge Portal on Archeology and Ethnography

The technology described in the previous section was used for building an ontology for a knowledge portal on archeology and ethnography. The system classification of archeological science proposed in [14] and developed today served as the basis for building the ontology of archeology and ethnography. The concepts of the system classification are used for building the classes of the ontology and the domains of their attributes and for creation of instances of these classes.

The ontology of archeology and ethnography includes four basic hierarchies: sub-fields of science, research methods, objects of research, scientific results (Fig. 2).

These hierarchies are founded on the following classes of concepts built on the basic concepts of the ontology of scientific knowledge:

Subfield of archeology and ethnography. This class is a root class in the hierarchy of scientific fields in archeology and ethnography. For example, subfields of archeology are *General archeology*, *Field archeology*, and *Ethnic history.*
Research method in archeology and ethnography. This class serves for description of research methods that are applied in archeology and ethnography. Such "traditional" subclasses as *Approach*, *Principle*, *Technology*, *Archeological methodology*, as well as a group of methods that came into archeology from other sciences (*Biological method*, *Physical method* and *Chemical method*), were defined as the descendants of this class.
Object of research in archeology and ethnography. This class is a root of the hierarchy of objects of research in archeology. It has such properties as description of an object, date of discovery, accuracy of dating, etc. This class has two subclasses *Material object* and *Immaterial object. Material object* is divided into subclasses *Archeological culture*, *Historical Person*, *Ethnos.* Successors of the class *Immaterial object* are *Artifact*, *Complex*, *Monument.*

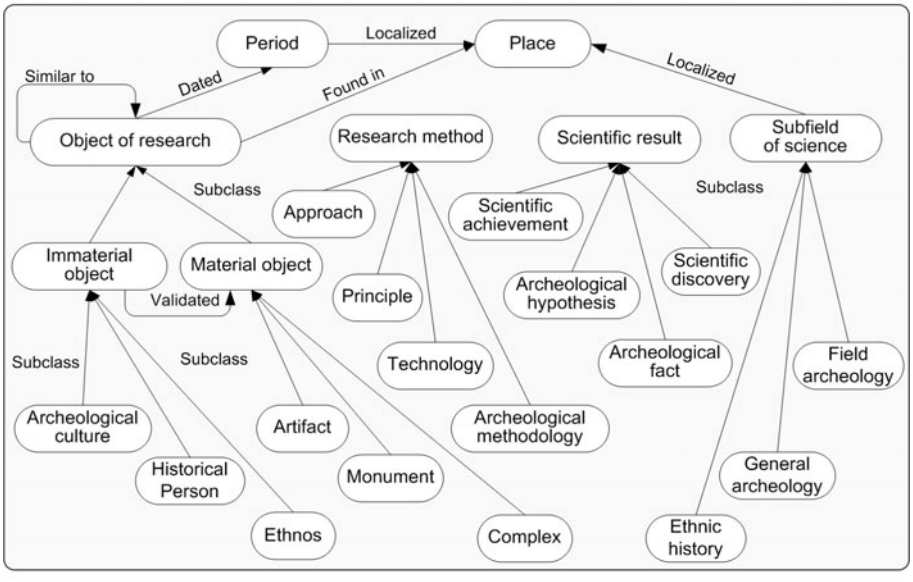


Fig. 2. A fragment of the ontology of archeology and ethnography

Scientific result in archeology and ethnography. This class serves for description of the results of research activity in archeology and ethnography such as discoveries, new laws, theories, historical facts, etc. The properties of this class are presented by such attributes as description of the result, date of obtaining and type of the result. The class includes the following subclasses: *Archeological hypothesis*, *Archeological fact*, *Scientific achievement*, *Scientific discovery.*

In addition to the above listed classes built on the basis of meta-concepts of the ontology of scientific knowledge, the classes *Archeological period* and *Place* were inserted in the ontology. The class *Archeological period* is specific for a historical science. It serves for dating of objects of research. *Archeological periods* constitute a hierarchy of nesting and historical sequence and are described by the time domain. The class *Place* serves for pointing to the location of an object of research or an organization, as well as for gridding (fixing) the subfields of science.

All hierarchies described above are interconnected with each other and with the classes of the ontology of research activity by means of associative relations. One part of these relations is inherited from the basic ontologies, the other part of them contains specific relations of a given subject domain.

So, the hierarchy of methods of research is connected with the hierarchy of objects of research by means of the relation “Uses” and the hierarchy of scientific results, that serves for typification and description of the results of research activity, is connected with activity by means of the relation “Result of”.

The hierarchy of scientific results is connected with publications where scientific results are described by means of the relation “Describes”. The objects of

research are interconnected with each other by means of the relation "Similar to" that defines the degree of similarity of objects and by means of the relation "Materially validated" that connects an immaterial archeological object of research with a material object of research that validates its existence.

Connection between the hierarchy of scientific fields and the methods of research in use is provided by means of the relation "Uses" and the objects of research are connected with scientific fields by the relation "Study". A chronological and geographical location of a scientific field can be defined by means of the relations "Dated" and "Localized".

Note that a complete ontology of the portal on archeology and ethnography includes the ontology of archeology and ethnography and the ontology of research activity described in section 3.2.

7 Conclusion

The paper presents a technology of ontology building for knowledge portals related to humanities. This technology includes the methodology of ontology building, the ontology description language, and ontology editor that supports ontology construction. A specific feature of the methodology of ontology building is the usage of the basic ontologies defined in part 3.2 as the basis for constructing an ontology for a knowledge portal.

The technology was used in the development of an ontology for a knowledge portal that provides semantic access to systematized knowledge and information resources related to archeology and ethnography (the Russian version of this portal is available at `http://www.sati.archaeology.nsc.ru/classarch2/`). At present this technology is used for building an ontology for a knowledge portal on computational linguistics.

References

1. Zagorulko, Y., Borovikova, O., Bulgakov, S., Sidorova, E.: Ontology-based approach to development of adjustable knowledge internet portal for support of research activity. Bull. of NCC. Ser.: Comput. Sci. (23), 45–56 (2005)
2. Gruber, T.R.: Toward Principles for the Design of Ontologies Used for Knowledge Sharing. International Journal of Human-Computer Studies 43(5-6), 907–928 (1995)
3. Guariano, N., Giaretta, P.: Ontologies and Knowledge Bases. Towards a Terminological Clarification. In: Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing, pp. 25–32. IOS Press, Amsterdam (1995)
4. Ushold, M., Gruninger, M.: Ontologies: Principles, Methods and Applications. Knowledge Engineering Review 11(2), 93–155 (1996)
5. Ushold, M., King, M.: Towards a Methodology for Building Ontologies. In: Proceedings of the IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, Quebec, Canada, August 1995, pp. 6.1–6.10. AAAI Press, Menlo Park (1995)

6. Benjamins, V.R., Fensel, D.: Community is Knowledge! in (KA)2. In: Gaines, B.R., Musen, M.A. (eds.) Proceedings of the 11th Banff Knowledge Acquisition for Knowledge-based Systems Workshop, KAW 1998, Banff, Canada, April 1998, SRDG Publications, Department of Computer Science, University of Calgary, Calgary (1998), http://ksi.cpsc.ucalgary.ca/KAW/KAW98/benjamins1/
7. OWL Web Ontology Language Guide (2004), http://www.w3.org/TR/owl-guide/
8. Protege, http://protege.stanford.edu/
9. Zagorulko Yury, A., Popov Ivan, G., Kostov Yury, V.: Subdefinite Data Types and Constraints in Knowledge Representation Language. In: Joint Bulletin of the Novosibirsk Computing Center and Institute of Informatics Systems. Computer Science, vol. 16, pp. 153–170. NCC Publisher, Novosibirsk (2001)
10. Lenat, D.B., Guha, R.V.: Building large knowledge-based systems. Addison Wesley, Reading (1990)
11. Swartout, B., Ramesh, P., Knight, K., Russ, T.: Toward Distributed Use of Large-Scale Ontologies. In: Proceedings of Symposium on Ontological Engineering of AAAI, Stanford, California, pp. 138–148 (March 1997)
12. Stojanovic, L., Motik, B.: Ontology evolution within ontology editors. In: Proceedings of the OntoWeb-SIG3 Workshop at the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW), pp. 53–62 (September 2002)
13. Using Dublin Core, http://dublincore.org/documents/usageguide/
14. Kholushkin, Y.P., Grazhdannikov, E.D.: Systemic classification of archaeological science (Elementary introduction in science of science), Novosibirsk (2000) (in Russian)

Methods and Technologies of Digital Historical Factography

Alexander Marchuk

A.P. Ershov Institute of Informatics System, SB RAS
mag@iis.nsk.su

Abstract. In this research work a special model is proposed for fixation and structuring of historical facts. The model is based on factographic principles of collection of formally specified data and information. The structuring model includes several classes of instances such as: persons, organizing systems, geographic systems, documents and some other. Instances are connected by simple or complex relations. Proposed approach is oriented to processes, upcoming in time and space, it allows to create databases, which do not become obsolete and do not depend upon time, place and position of perceiving person. Factographic approach can be used in a set of such applications as: building of digital archives, museums, historical encyclopedia and directories. Same ideas can be efficiently implemented in small systems for office work organizing. New possibilities in use of a structured field of facts for analytics and knowledge extraction are under investigation. Specific features of the proposed approach are: the main concept stems from the Semantic Web, a new ontology of non-specific entities is proposed, strong orientation on distributed databases, use of the idea of information space of documents. This approach was partially implemented in applied projects of A.P. Ershov Institute of Informatics Systems.

1 Introduction

The data collected and accumulated to accomplish a specific task are most expensive in modern information industry. The problem is that a great number of sources are badly compatible and informational units describing the same phenomena are widely duplicated. To take an example, as users of a variety of services and Internet portals we are authorized in different data bases with our personal information, however any information system to come next would offer to enter a similar set of data. Personal data are said to be private, the legislation limits distribution of personal information. That is true, but the same thing concerns the data which are not covered by limitations and in fact are public and free for distribution. These include the data on companies, geographic objects, and open documents. It is quite evident that the data located in one reliable source would be much more interesting, relevant, and valuable. We treat it as the problem of *common information field*.

Some achievements aimed at constructing a common information field have been made. UDDI [1] system allows registry of information services, RSS is used

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 217–231, 2011.

to unify a news flow, Dublin Core [2] is used to describe a broader range of information resources, MARC, CIMI [3] standards are used for special purposes of such recourses, etc. The problem is due to the lack of a holistic approach to manifold data structuring which would allow a person or a computer agent seek for a reliable information required and use it in a particular task.

Another aspect of a number of problems approached in the present research is the accuracy of data arrangement. In a number of cases the available database schemas appear to have been developed without any principles, except for the principles concerning normalization and some other evident aspects. The most common mistake of database programmers is that they see only a current state of a problem. For example, a database of a company staff may include only a current stuff and a date of admission. And how about former employees? Otherwise the database could have a historical reference. Such modification could be quite easily fulfilled just by entering a dismissal date. In fact, it is not that easy. A number of new problems may appear preventing the database from being historical. Suppose, an employee used to hold more than one position, or some femal employees changed their surnames after marriage; everything may change from passport data, telephone numbers, addresses, academic degrees to a company structure and name.

The approach proposed is referred to a factographic or fact-based approach. It should be noted that it is based on recording facts and not on deletion or editing. Our aim is to formulate the principles of manifold data structuring, with all the facts previously recorded being true irrelative to time, geography and other factors of their reproduction. The fact base can be changed only by adding new information units (statements). To bring a simple example, it is incorrect to specify the age of a person as a number of years, though it is correct to indicate a date of birth. The aim formulated is ideal. Currently, in implementations there are tools to change and to delete information fields and links.

This approach is more natural for a historical profile as it fixes the facts referring to a long period of time and often to a vast geographic distribution. The basic structuring principles and techniques have been developed by database formation and maintenance at the Programming Department of Novosibirsk State University with a very high dynamic performance and annual rotation of students from one category to another. The present task is referred to systems for office work organizing and its aspects have revealed key problems of the factographic approach. The approach has lately been developed in a number of historical projects, although it is of current importance to correlate the historical material with the present day facts (office work organizing) as tomorrow the present events will be a historical fact.

Returning to considerations of information privacy, the approach proposed makes the problem more urgent. In fact, having and processing a full data set on different aspects of persons of a particular social group care is to be taken not to let this amount of information units fall into criminals' hands. To illustrate, the following private information can be published: date of birth, place of employment, address, communication address, affiliation to other persons,

activities attended, programs of these activities, colleagues' names and their telephone numbers to contact and to obtain some additional information, etc. And I do assure you that a similar information on public figures can be obtained in open sources, the data brought together being a potential hazard. That is why our group prefers to consider the past and, alas, the people passed away.

Another concern of the approach considered in the present research is to develop distributed systems. In fact, a host of information resources are developed and supported by independent teams having their own objectives and preferences. As a rule, they wish to manage their results at their own discretion without following any external processes. The most evident result in this case is a self-contained development with its essential drawbacks: first, professional implementation solutions are not always used; second, the information collection does not enlarge a common information field; third, after the project is expired the information resource may result in failure. Our approach allows solving the three problems stated without changing the autonomous status of work. The approach offers a number of methods and techniques that could be used independently and professionally and developed due to further changes in basic programming systems. The approach allows dividing the information field into private and public information; it has both mechanisms to expand database schemas to take account of a specific local objective and tools to adjust a basic solution to the application domain and users' preferences. If the project is completed, the database developed can be involved in a single-source data depository and keep on forming a common information field.

2 Common Information Field

At present there are a number of approaches to fix history-oriented facts. These are databases, World Wide Web, systems describing information resources based on metainformation, and digital libraries. A new direction in information technology such as Semantic Web (SW) has recently been developed [4].

The drawbacks of relational databases used for fact fixation can be referred to a centralized storage and processing, difficulties in maintenance and modernization inconveniences to work with specifications and dictionaries. WWW technology is not generally aimed at formal approaches to data structuring, it is supposed to use information, to do the search and navigate. Substantial improvements have been made in structuring systems based on metainformation and electronic libraries. The shortcoming of the approach is a 'shallow' structuring mostly used for greater amounts of homogeneous information and usually not producing databases of manifold elements. In contrast to the approaches mentioned SW technique allows formalizing essential knowledge of data implication and using this knowledge to develop information systems of new generation as well as it allows developing distributed systems based on a full-scale database. At the same time the data field can easily be used to make both WWW-interfaces and requests from computer agents.

Semantic Web has been developed as a basic approach for a wide range of factographic systems. It is conditioned by the fact that the approach can use

disconnected or loosely connected information sources. Besides, the formalisms existing in SW and the adjusted standards are appropriate to the objectives of a distributed field of manifold information. Working with the data specified by OWL [5] tools, a logical conclusion is likely to be made for a number of important cases and other artificial intelligence technologies can be applied. Programming use is being currently studied in bottlenecks, fuzzy logic, semantics and data clustering.

The approach proposed implies models in which entities of the outer world are attributed to information units and relation between the entities are realized by both a direct reference and a composite construction of a specific type. Specification of such models realized as the ontology makes it possible to develop universal processing software, to use specification to make the data more accessible, to realize multilingual information systems, to analyze whether the data are full and correct. Being 'in the centre' of SW technique RDF [6] uses the basic concept of semantic network of single statements and groups of statements. Statements are united in a single graph by combining equally identified information units (items). This allows uniting RDF models stored in different locations resulting in a naturally distributed information field.

Consider the look-and-feel of a distributed information field. In details the information field can be viewed as a set of RDF documents logically associated with each other by a common identification of objects and relations being equally treated. The term 'publication' is used to describe the document attributed to a fixed URL and 'seen' from the Internet. Possible details are associated with the limited visibility for different categories of users and an indirect access to a document. The latter means that a relational database in a database application may have a particular interface transforming a database or its part in a required RDF document. The documents published are opposed to the unpublished ones, i.e. the access is authorized only for 'friends'.

Thus, any Internet agent may have an equal access to common published data and a local access to personal documents. From the amount of regularly built documents any agent can form a database as a semantic net, or in some cases as a system of relational tables. The use of such database may result from the functions of the agent collecting the database. As a result, a common scheme of information system organization with proper information content based on a common information field has been made.

A common field of facts will be useful only if the problem of 'understanding' is solved, i.e. if any document with common data uses data structuring relevant to the one used by a specific agent (specific information system). Yet this is not enough. All the objects of different databases denoting equal entities should also be united under merging databases. We refer the first objective to the ontology compatibility, and the second one to the problem of identification.

In its 'rough' version the common information field constitutes a number of RDF documents structured according to the unified ontology and identification is made by basic RDF tools through equal entity identifiers. In its 'gentle' version the data from downloaded documents are restructured according to different

ontologies and the problem of identification is solved by a certain algorithm approach. It is to be noted that if the information is not complete the identification algorithm may not give any good result in a number of cases. In fact a world-wide practice of persons' identification requires a full name of a person, his place and date of birth, the latter being often omitted in databases. It should be remembered that a name could be spelt in different languages and be changed, etc.

To sum up, a 'gentle' approach providing data compatibility can be more practical. Our approach aimed at ontology compatibility requires that different documents of the common information field should use only extended ontologies called basic ontologies. Quite appropriate would be the ontology on libraries of reused components for specialized programming complexes. If these libraries are great in number and not connected to each other their use would be quite constrained. Yet if the libraries make a hierarchy of a layered extension, the extensions being independent (orthogonal) can be used by special programming complexes to fulfill their own objectives.

We solve the problem of identification in the following way. Steps are taken to do the identification on the earliest stages of a particular data domain development, for example, on the stage of primary data input, and a manual data input in particular. It can be provided by 'total' databases of standard entities such as persons, companies, cities, and etc. Although in real situations when data are written by independent processes algorithmic solutions and the corresponding tools are necessary for the systems of editing and manipulation.

The question is whether an entire model (database) should be developed in each processing server. Is it possible to distribute both data and data processing? There are likely to be some approaches to this task but in general it is not covered either mathematically or technically and is not to be discussed in the present study.

3 Basic Ontology

The key focus of the approach is ontology of non-specific (basic) entities. The term 'non-specific' is used in contrast to 'specific' entities. In fact, any information system has its own object and specificity. In addition to specific entities such as account numbers, transactions and account balances it is often necessary for the ontology to have such classes of entities as persons, companies, locations, communication addresses, i.e. the entities resulted from the worldwide culture and modern civilization deprived of a specific character and well understood by system developers and users.

Unfortunately, such ontology of non-specific entities has not yet been developed as a universal standard or a recommendation, although it is being traced in many standards like (DC, FOAF [7], CIMI). According to a number of requirements we have found and developed the basic ontology which is currently being used in a number of projects.

Basic ontology has been developed [8] within a long-term research work with real projects. And it should be emphasized that it has not yet been finished.

Incoming changes always demand that the data accumulated should be regularly transformed, although the methods of abstract programming based on explicit data specification allow maintaining most of program codes and interfaces under such changes.

To develop ontologies, basic ontology in particular, a number of principles have been defined.

1. **Description of real-world facts**. When developing a common information field it is reasonable to follow a factographic approach to the world model construction. It means that real entities are to be the object of fixation in a database and the information of such entities is to be objective and independent of a particular data domain these facts are used for. For example, if we develop a historical database, describing a person we are interested in facts like his date of birth, academic background, place of employment and address. We would not fix such subjective factors as evaluation of his success, popularity and authority. These factors may come out in a particular outlook on historical processes fixed by a particular information system that, as it has been mentioned above, can make a wide use of facts of the common information field.
2. **ER model as the base of ontological construction**. The conventional Entity-Relationship model is natural and effective for this approach. In fact, describing entities and entity attributes we construct a field of independent information units. Describing relations between the entities we construct a coherent database with one object being able to 'reach' another, which corresponds to the universe arrangement. If the database falls into independent parts it means that data are incomplete or badly structured.
3. **Detailed description balancing**, and minimization of national and cultural differences. It is impossible and unreasonable to describe the world in every detail. Therefore a detailed description should be limited to a certain range. The same is relevant to a set of entities and relations. Detailed description is also limited by a capability to equally understand the input data and to distinguish between different groups of developers and users. For example, in historical retrospective view types of management structures may be quite different from country to country as well as in one country. Executive positions, territorial administrative division, documents, relatives etc., may also have different names and would not be understood and processed when analyzing data on different time periods and countries.
4. **Special realization of attributed relations**. One of the most common mistakes made by ontology and database scheme developers is that they ignore possible attributes for a particular relation. In ontologies developed by OWL tools wrong solutions are made when a direct reference is used as a relation by defining object attributes. For example, the relation 'work' between a person and a company is often treated as an object attribute. This is wrong. A person establishes relations with a company at a particular moment of time and can break them off at a different moment, so that the relation 'work' may include some other attributes such as his position, look Figure 1.

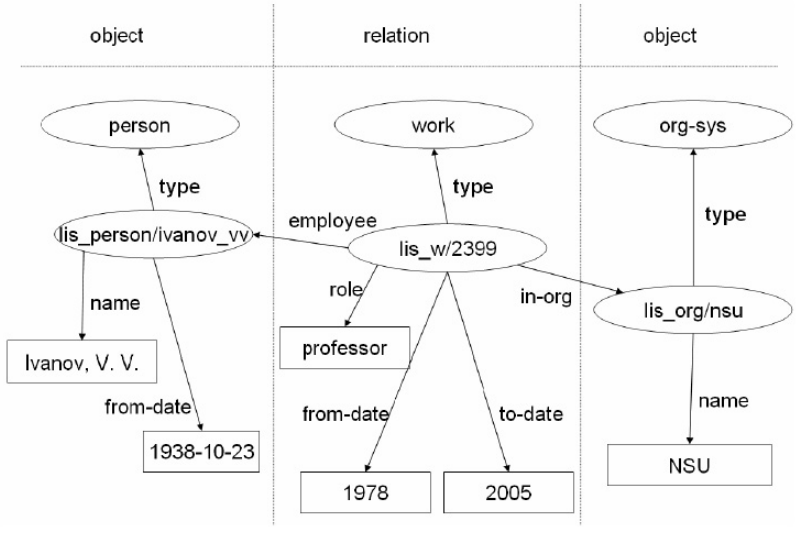


Fig. 1.

5. **Direct fixation of context information**. People are used to communicating in a particular context. For example, '*Putin said. . . '*. It is quite evident that the statement is attributed to a particular person among tens of thousands of Putins, to a recent event, and beyond it there is likely to be a particular activity, place and some other context attributes. In a great number of cases a certain context is available in formal data specifications making it difficult to group the data obtained from different implicit contexts. Context information is often used to group information units, e.g. lists of persons working or living in the same place, role grouping (teachers-students, doctors-patients, employees grouped according to staff organization hierarchy of a particular company, etc.).

The analysis shows that grouping and hierarchy relations are not effective for data structuring and can make data access quite sophisticated. Grouping is a secondary tool and it should be applied to data features and relations written in a database.

6. **Reducibility of basic ontology to the system of simple relational tables**. According to this principle an ontology is to be constructed in such a way that data structuring and storage medium should comprise both RDF documents and simple relational tables and they should be in good agreement with each other from the realization standpoint. The principle is of great importance since it makes constructed RDF data compatible with the existing relational data and allows use of relational database applications as a medium to support semantic net models. Such compatibility seems to be quite natural: each class of entities is associated with a table where entity identifier is a primary key; attributes (DatatypeProperty) change into typed columns and direct references (ObjectProperty) become foreign key columns. When constructing a basic

ontology such compatibility makes a set of solutions substantially limited. First, to define an entity type multiplicity has to be neglected and ontology requires compulsory entity typification. Second, DatatypeProperty and ObjectProperty attributes are not inherited. Third, ontology does not use 'complex' RDF constructions like blank nodes, containers, lists, reification. Fourth, multiple values are not to be used in columns of relational tables. One can do so only if he chooses 'correct' relations and controls directed arrows in the attributes associated with different entities.

Four classes of entities have been defined for the basic ontology under view: persons, organizing systems, documents, and geographic systems. The basic entities are associated with a number of relations schematically shown in Figure 2.

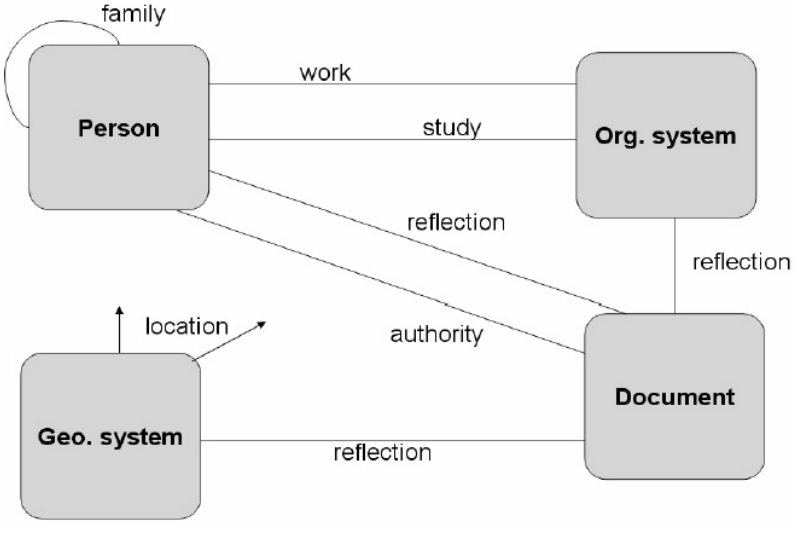


Fig. 2.

Organizing system is viewed as a (temporary) community of people aimed at achieving particular results. This category includes companies and teams, associations of people and companies. Geographic systems imply cities, countries, and other places of activity. Documents involve a wide set of exhaustive media (books, papers, files, pictures, etc.) representing a fixed information content. Ontology defines the following relations between the stated entities. People are described only by family relations, persons and organizing systems may be related by a 'participant'. 'Education' has also been defined as an important relation. It has become a tradition to show the education background in Curricula Vitae. Other classes of documents and entities are related by reflection, i.e. such entity is depicted or mentioned in document content. Besides, if a person is an author of a document he is related to 'authorship'. Geographic systems represent a place for a particular point in a database and they are marked as 'location'.

Among other classes of entities belonging to basic ontology there are collections and archives. A variety of unary and binary (complex) relations such as allied organizations, titles, ranks and rewards, communication addresses etc., are shown in the Figure 3. As an example we show how to write an e-mail address for a person.

```
<email rdf:about="em134983">
  <user rdf:resource="mag" />
  <email-code>mag@iis.nsk.su</email-code>
  <from-date>1990</from-date>
</email>
```

Fig. 3.

It is quite clear that an e-mail address is attached to a person without limiting the number of addresses and 'closing' an expired address. The latter can easily be done by adding <to-date>YYYY-MM-DD</to-date> showing a completion time of 'e-mail' relation.

In basic ontology a wide use is made of subclasses to indicate specific cases and some subclasses are marked as abstract (by tools alternative to OWL formalism). It should be noted that using mechanisms of classes for a complex relation the term Thing defined for OWL root class does not seem to correspond to the term 'relation'. For example, 'participant', 'location' and some other relations are complex and cannot be identified as 'thing'. In our ontology a root class is abstract and be identified as 'entity'. For classes of basic entities corresponding to the term 'thing' we use a sys-obj-system object class. This abstract class is inherited by persons, organizing systems, etc.

Quite innovative for basic ontology are the terms 'dating' and 'naming'. First, dating. In root classes time limit is directly defined as a basic attribute of system objects and relations through from-date and to-date (DatatypeProperty) indicating time stamps for start and finish time of entities and relations. In fact, this appears insufficient. Working with data it is often difficult to indicate exact values of these attributes. In contrast, expressions like 'the entity (relation) has started to a certain date' or 'the entity has not (or has already) finished, could as well be used. An approximate date could sometimes be indicated which is not the same as indicating an exact date.

Some solutions of the stated problems can be found in unary dating relation. To indicate the entity a certain date having a qualifier and exact definition is defined by separate dating. Qualifiers can be marked as before-date, from-date, in-date, to-date, after-date, i.e. one of five meanings of marking a time axis containing an interval for a specific case. For example, Figure 4 shows that we mark a time point '2006', which is accurate within a year, as a finish time of the entity having da298345 qualifier.

Now we consider 'naming'. System objects have a standard attribute 'name' denoting a name of an object. As in the case described above it is yet insufficient. As a result, a unary relation 'naming' indicating that a synonym can be used

```
<dating rdf:about="da298345">
  <referred rdf:resource="em134983"/>
  <date>2006</date>
  <dating-specificator>to</ dating-specificator>
  <dating-accuracy>1-0-0</dating-accuracy>
</dating>
```

Fig. 4.

as a name of a system object. Besides, we now have an opportunity to make multiple indication of an object and naming can be attributed, e.g. it is possible to show the time limit within a particular name has existed.

4 Data Input and Editing, Information Visualization

The systems based on formal specifications are very effective for advanced data abstraction. A programmer does not think in terms of persons or cities, but he does so in terms of mathematical models associated with the real world by means of ontology that he also considers as a mathematical object. Such level of abstraction is often difficult to follow but it is possible and necessary for basic operations: visualization, editing, search, and navigation. The programming tools obtained are getting universal, and to use them one can easily specify ontology and some other specifications. The system developed can 'talk' to users in terms of a specified knowledge domain and 'speak' any language relevant to users.

Let's consider RDF as a mathematical object and some of the tasks a user seeks to solve.

RDF model is an oriented graph whose nodes represent entities (treated as subjects in standard terminology) and string constants. Arcs represent 'entity-entity' and 'entity-constant' relations. When necessary we outline some other properties of the model – typification, identification, compatibility with formal specifications (ontology). Let item be a typed entity of the model. If we focus on a particular item we can see its canonical presentation, Figure 5.

Note that we mainly focus on typed models, i.e. the models with defined entity classes, properties (DatatypeProperty, ObjectProperty) and their possible use. We also focus on ontologies with cardinality of output arcs equal to one. The basic ontology described in the previous section possesses the stated properties.

As it can be seen, focusing on a particular item specifies a subgraph neighboring to this object. It is clear that item attributes together with the item neighborhood give information essential for its characterization. Let such subgraph be an information portrait of an item (entity). For example, under the basic ontology, if we focus on a particular person we may obtain information on his age, relatives, academic background, job, address, participation in different activities, communication addresses, etc. Sometimes it would be useful to 'look' at more than one arc in the object neighborhood.

If the model uses metrics, such as weighted total of arcs between graph nodes, the task of information portrait construction would be just to mark a ball of

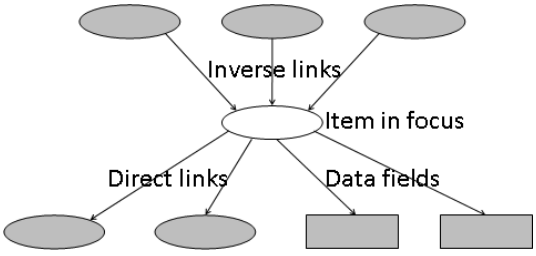


Fig. 5.

a particular diameter in a graph with its centre in the item of a target object. Such geometrical subtask is not the only interesting organization. If we specify two nodes (items) we can develop a certain neighborhood of points whose total distance to focuses would not be more than a given value, i.e. we can build an ellipsoid. The construction is quite effective as it allows formation of the information portrait of an item with respect to the item like '*What kind of object is it and how is it associated to me?'*. There are some other ways to specify a subgraph based not only on metrics and arc weight ratios but matching specified note weights. It is not difficult to construct a shell for N-specified points and interpret it as developed subject. Such constructions can be effective both to extract information contents, to make references (history) and to organize key and object search.

RDF net structures have some peculiarities. It concerns information visualization, search and editing. And all the three processes can be mixed within the same operation. Let's start working with search, formulate a search request of any form. A search request may provide candidate items corresponding to request criteria. If a request is too complex and information field is not sufficient there may be a number of candidates. The next step is to visualize a range of candidate items in such a way that a user could find the one required. Visualization of item lists is usually based on visualization of a separate item; the problem is how detailed a portrait could be. After the searched item is specified a user may be interested in its detailed information portrait. From the peculiarities of RDF modeling it becomes clear that the information portrait consists of a number of attribute values and neighboring items. A number of attribute values can be easily realized as sets of pairs like 'attribute name-attribute value' and represented in a conventional form as a data table. Sets of associated items can be visualized by the afore-mentioned tools.

Difficulties may appear during editing when an operator should constantly solve two tasks both being at least unusual to this operator in the context of editing. The first task is to search and to identify information units and the second is to associate one object with another by simple and complex relations. Why are these procedures so unusual and difficult for the operator? In fact, while inputting and editing data the most stable and usual constructions for input and editing are entities for single objects and a table for multiple objects. For a visual

interface these are either form fields or table rows. Reference editing is usually not provided. Thus, the problem is how to specify naming of a particular item with another item of the database. Let's consider a task for 'direct' references; in a graph model it corresponds to arcs coming from one item to other items. For example, (ObjectProperty) location is to be associated with a particular reference by means of visual interface. What should the input reference interface look like? And the interface is likely to be 'inside' the location not to contaminate the portrait of the edited item. There are a few solutions applied in different systems.

The easiest way is to change a set of candidate items into a visual interface element like ListBox or ComboBox. Despite the fact that this method appears to be most familiar to a user, there are a few drawbacks limiting its application. First, if a user does not input additional information, a list of candidates to be associated could be too large, which makes this method unpractical. In this case ComboBox is more preferable as it allows input of search information. Second, the item to be associated may not be found in a database. In this case a referenced item requires a special initiation form. Third, to define what item of the list is searched for one should study candidate details, which is also unusual for editing.

Another method is to apply a search interface to a particular edited location, and directly to the edited box. In this case, first comes a form to formulate a search request and then the already mentioned list of items to choose from; there is also a 'button' to build a new item, if the search fails. A new item is built according to the existing context and the information in search request. Such universal solution may be quite sophisticated, if a type of the attached item is different. In this case the type is specified by a user, a search request form is generated and only then we can obtain a list of candidates. One of the advantages of this method is its universality and relative compatibility with standard realization schemes of Web interfaces and window applications.

The third method implies using drag and drop technologies to build direct references between items. Using this method we can find and generate any associated item by an independent interface, then 'drag' and 'drop' it to any box or directly to the current item. The advantage of this method is use of basic concepts of structuring models: there are entities (items) and relations, and editing implies search and input of items, editing their attributes and associations. Although, a user may not sometimes be familiar with a model of data editing and should have a special knowledge of the ontology used.

The text box model could be more natural to a user. For example, the field 'address' would allow some text with minimal elements of formal syntax, say, *Novosibirsk City*. The system could understand this phrase with dictionaries of terms and abbreviations and find a proper item in a database to be used as a reference, and if it failed it could enter a new element to be referenced. It is evident that identification may be alternative, but a user may always be warned and given an option to choose. This scheme requires a highly developed database with basic categories of entities, indication of a currently edited context and convention systems regulating use of function words and abbreviations, or a

highly developed analysis system for natural language texts based on algorithms of artificial intelligence. Besides, the scheme is not universal. For example, if searching involves documents (pictures, etc), it would hardly be based on a document name. Verbal identification of some activities and organizations may sometimes be difficult.

The above-mentioned peculiarities of editing as applied to semantic nets allow to conclude that there are no suitable models for a user to do search and editing. To obtain the same result alternative methods may be taken account of.

Figure 6 screenshot shows an editor, designed for work with RDF models.

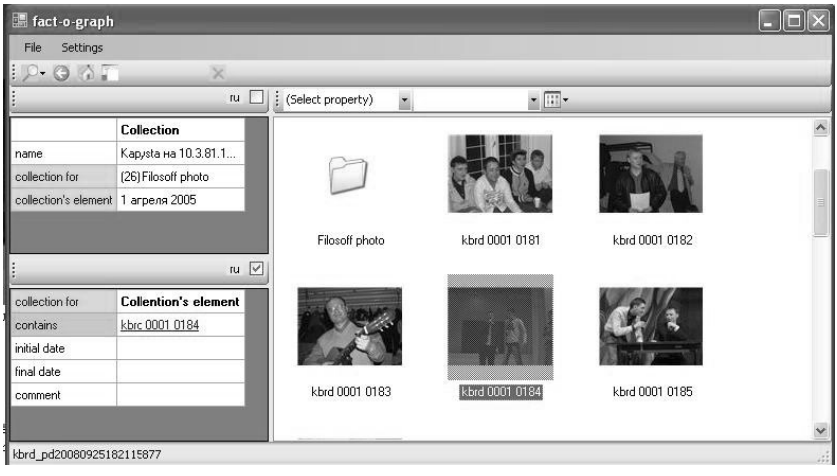


Fig. 6.

Quite often the problem is not the absence of information, but its excess. For example, in a database a certain person may have thousands of pictures and other documents he is referred to, hundreds of documents written by him, tens of conferences he attended, etc. Such integrated information portrait makes it difficult for a user to find the required information and relations between information units. Thus, if a user wants to find a telephone number of a person, he might find it quite difficult, when a previously mentioned list of items referenced to his request is badly organized.

In fact, consideration is taken of visibility restriction which is specified statically in a particular interface setup, and dynamically for the work with a particular object. We have tried two approaches to solve this problem. The first approach is to define sets of classes and features which are currently not being considered. These sets are specified by groups called subjects. For example, the subject 'documents' includes several classes of documents and a number of relations and features associated with these documents. These include authorship, reflection, and publication. Data interface includes setup lists of subjects, and the subjects to be browsed and edited as well as those, which are not used, can be marked.

Another method deals with information value of items. This factor is quite subjective and makes it possible to sort item lists according to a certain value, to keep the most valuable information in the information portrait.

5 Conclusion

An approach to build electronic archives has been formulated. The aim of this approach is to generate electronic images of archiving units, to build and to use databases of conventional entities (persons, companies, geographical points, events), to 'bind' archiving units to database elements, to build the information system allowing navigation and search in a database.

Historical information systems have been studied and developed by the Institute of Informatics Systems of the Siberian Branch of the Russian Academy of Sciences for 10 years. In particular, the electronic archives of A.P.Ershov's documents have been developed and used for several years. As a result of research, the basic principles of building historical factographic systems have been formulated: the system should reflect facts of the real world; in this respect formal systems defining the nomenclature of reflection objects and basic data features through specifications have been employed; distributed information systems should be developed; databases should be context-independent; a wide use should be made of international standards; Web-clients (browsers) are to be system interfaces for people, and Web-services should make system interfaces for program agents.

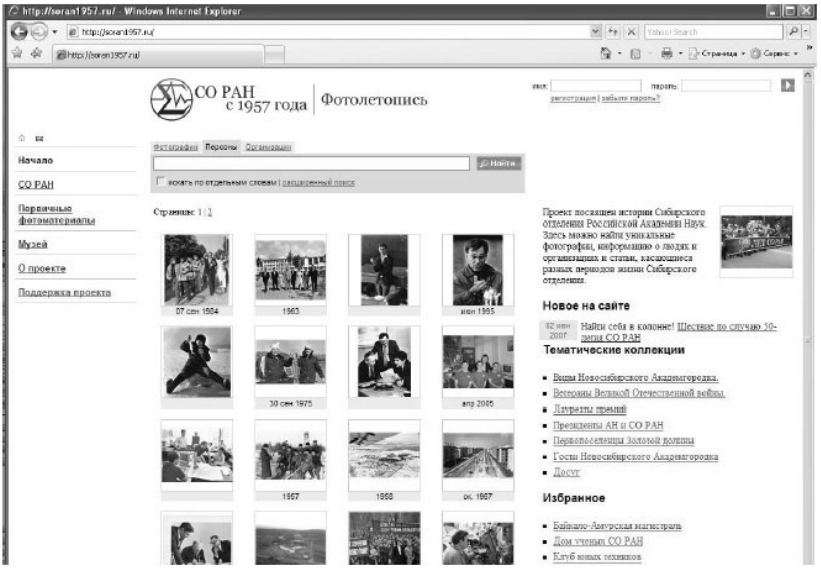


Fig. 7.

The most significant recent project was the Photographic Database of the Siberian Branch http://soran1957.ru first launched by the 50^{th} anniversary of the Siberian Branch of the Russian Academy of Sciences. The database now contains about 13.000 photographic documents, comprises the period since 1957, and includes information on 4.000 persons, 2.000 organizations and events, and all the members of the Russian Academy of Sciences since its foundation.

At Figure 7 a screenshot of the public interface of Photographic Database is shown. The database is a software-hardware complex that comprises scanning and other types of processing photographic documents, guarantees information security in case of an unauthorized access, represents effective tools for editing, operational and public interfaces. Public interface of the database was first launched in 2007 and within the first three months it had about 20.000 web visitors who browsed over 600.000 pages.

The Institute of Informatics Systems, SB RAS will keep on working at the Photographic Database improving the technology, supplying it with new data and detailed description of documents, and making it more accurate. On its platform a number of new corporate and private databases comprising fact bases, documents, photos, video and audio documents, descriptions of people, organizations, events, and projects, will be developed.

References

1. Online community for the Universal Description, Discovery, and Integration OASIS Standard, http://uddi.xml.org
2. The Dublin Core Metadata Initiative, http://dublincore.org/
3. The CIMI Profile. Release 1.0H. A Z39.50 Profile for Cultural Heritage Information, http://www.cimi.org/public_docs/HarmonizedProfile/HarmonProfile1.htm
4. Tim, B.-L., James, H., Ora, L.: The Semantic Web. Scientific American 284(5), 34–43 (2001)
5. Web Ontology Language (OWL), http://www.w3.org/2004/OWL
6. Resource Description Framework (RDF), http://www.w3.org/RDF
7. FOAF, http://www.foaf-project.org/
8. Marchuk, A.G.: Distributed digital archives, libraries and databases. Preprint 122, A.P. Ershov Institute of Informatics Systems, pages 25. SB RAS, Novosibirsk (2004) (in Russian)

Establishment of Taxonomic Relationships in Linguistic Ontologies

Natalia Loukachevitch

Research Computing Center of Moscow State University
Leninskie Gory 1/4,
119899, Moscow, Russia
louk@mail.cir.ru

Abstract. The paper describes typical problems arising in the course of establishment of taxonomic relationships in ontological resources. The situations in that the taxonomic relationships get mixed with relationships of other types are very diversified. In the paper we will discuss the ways for revealing the mistaken taxonomic relationships. Our consideration is based on the experience of development of large linguistic ontologies such as the RuThes Thesaurus and the Ontology on natural sciences and technologies.

Keywords: Taxonomic relations, ontology, linguistic ontology.

1 Introduction

In various computer resources and applications, such as ontologies, artificial intelligence systems, object-oriented programming, information retrieval thesauri and many others, the relationships between the classes and subclasses of concepts are the key ones. Such relationships are usually considered to be hierarchical (partial order relations), transitive and having inheritance: attributes and relations of a higher-level concept are inherited to a lower-level concept.

The relationships between classes and subclasses of concepts may be called differently depending on an application domain: taxonomic relationship, broader term or narrower term relationship (in information-retrieval thesauri), isa relation (in artificial intelligence), hyponymy and hyperonymy (in lexical resources). Further in this paper we will refer to this relationship as the "taxonomic relationship".

Defining the taxonomic relationships developers of resources can use, explicitly or non-explicitly, the classical diagnostic language expressions. For example, if concept X is a specialization of concept Y, then it may be said that "X is a Y", "X and other Y", "X is a kind of Y" [1]. Word or term definitions in dictionaries also look like similar diagnostic expressions: "X is a Y that…".

However, in ontological terms the same expressions of a natural language may correspond to significantly different relationships between the entities of the external world, including those possessing quite other properties [2]. Therefore methodological guidelines on development of conceptual resources recommend additional checks before establishment of taxonomic relationships.

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 232–242, 2011.

The most known recommendation concerning establishment of the taxonomic relationships between classes (concepts) A and B is to check the following condition:

> *If class A is a subclass of class B, then each instance of class A is also an instance of B* [3, 4].

However, the situations in that the taxonomic relationships get mixed with relationships of other types are much more diversified and in development of ontologies and other conceptual resources such problems should be taken into account.

These problems are most essential for the developers of conceptual resources for natural language processing and information retrieval applications. In such applications the resources should, on the one hand, take into account the existing conceptual system of a language (group of languages) – this is reflected in special term "linguistic ontologies" [5]. On the other hand, conceptual relationships should be established on the basis of the conceptual, ontological analysis and not only utilizing the language diagnostic expressions.

At the same time it should be stressed that it is rather difficult to separate completely the ontologies created for other computer applications from the natural language. The ontological units often have language or mnemonic names, thus, additionally "provoking" the application of ambiguous language tests. Thus, Y. Wilks [6] asserts that the symbols in representation languages are fundamentally based on the natural language, that a representation language is a means of human communication with the inherent dynamics, polysemy and possibility of extended interpretation.

In any case developers of conceptual resources in various fields and for various computer applications may get into such a "trap" of language expressions. That is why it is important to describe the problematic cases of defining the taxonomic relationships and likely ways to reveal such deficiencies at the time of establishment. In addition, relying upon transitivity of the taxonomic relationships such local deficiency may turn into a serious distortion in the course of multi-step logical inferences.

The paper describes the types of problematic cases of the taxonomic relationships using as examples descriptions from the linguistic ontologies Thesaurus of Russian Language RuThes [7] and Ontology in Natural Sciences and Technologies ONST [8].

2 Criteria for Verification of Taxonomic Relationships

The criteria for verification of correct description of taxonomic relationships are connected with verification of the transitivity and inheritance properties.

The following rule is based on verification of transitivity of the taxonomic relationship: both the higher-level concept and lower level concept should relate to the same top-level concept such as *action, property, object,* etc.

So the standards and methodological guidelines on development of the information retrieval thesauri recommend application of this principle for description of the hierarchical relationships in thesauri [4, 9].

For example, in our practical work we faced such a situation: at defining of the taxonomic relationships in thesaurus RuThes the following chain of relationships was established:

RIVER – isa – *WATER BODY* – isa – *WATER OBJECT* – isa –
– *WATER* – isa – *SUBSTANCE,*

as a result of which it was found that all particular rivers belong to the top-level concept SUBSTANCE, which is not correct.

In this chain the most problematic is the relationship *WATER OBJECT* – isa – *WATER,* the substitution of which for some other type of relationship will resolve the problem (see section 5).

The second type of criteria for verification of the taxonomic relationships is connected with verification of the inheritance property.

This verification may be conducted relative to a particular pair of concepts.

For example, in dictionaries the *raisin* is defined as *dried grape* [10]. Does it follows from this definition that the taxonomic relationship between the concepts RAISIN and GRAPE could be established? In terms of properties inheritance the answer should be "No", because *raisin* does not possess many properties of *grapes* as fruits of some plant: it does not grow, ripe, it is not gathered. But, for example, in linguistic ontology WordNet 2.1 [11] *fruit* is described as a hypernym for *dried fruit*.

The inheritance property may be verified also on the basis of formal properties of concepts.

Thus, for analysis of correctness of the taxonomic relationships N. Guarino and C. Welty [12] suggest verification of the inheritance property to subconcepts based on such a formal property as "criteria of identity".

The idea of the criteria of identity of some concept is to define what does it mean when two entities representing the examples of one and the same concept are the same; how can the entity change maintaining herewith its identity; what properties are essential for maintaining its identity, etc. One can speak about sufficient conditions of identity, i.e. what conditions are used to determine the identity, and about the necessary conditions of identity, i.e. what does it mean when two entities are identical.

For example, two human persons should be identified as one and the same person if they were in one and the same place at one and the same time. Therefore, the condition of identity of human persons is the physical coincidence of their location by place and time.

If the assumed generic and specific concepts have different conditions of identity, it means that no taxonomic relationship may be established between them [12].

Further on we will discuss the particular types of incorrect description of the taxonomic relationships and show which criteria may be helpful to avoid such mistakes.

3 Confusion of Types and Roles

One of the most common problems in the description of the taxonomic relationships is confusion of types and roles within one hierarchy.

For example, the "type-type" relationships (*birch is a tree*) and the "type-role" relationships *(apple is food*) may be equally expressed by all diagnostic tests applied for defining the taxonomic relationships. The difference is that a birch remains a tree at any moment of its existence, while an apple may be used as food, may be used for other purposes, may be not used at all.

The most common mistake in the description of the subject field is location of role concepts as higher-level concepts over the type concepts. For example, as an employer may be a person or an organization, one can decide that concept *EMPLOYER* may be represented as a higher-level concept, while concepts *PERSON* and *ORGANIZATION* are represented as lower-level concepts [15] (fig. 1). However such representation describes the properties of entities not accurately, because not every person is an employer.

In many cases the analysis of role-type relationships may reveal violation of the basic principle of the taxonomic relationships on belonging of all examples of a lower level concept to a class of the higher level concepts (see Introduction) as it happens at incorrect defining of the relationship *PERSON* – isa – *EMPLOYER* . For the functioning of a logic inference mechanism such inaccuracy leads to the situation when for each instance of a concept *PERSON* a system will make a conclusion that this is an instance of the concept *EMPLOYER*, which is not correct in general.

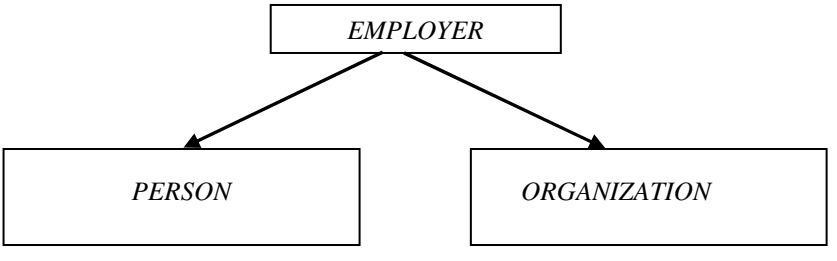


Fig. 1. Confusion of type and role concepts in the same hierarchy

In other cases the problem is not so obvious. For example, at defining of the relationship *APPLE – FOOD* the developer of an ontology may take into account the specific features of the modeled subject field in which all or the greater part of apples may be considered as food.

In subsequent sections we will consider how role concepts are defined, how to describe knowledge about main roles of a concept remaining within the framework of simple knowledge representation models and do not violate the principles of establishing the taxonomic relationships.

3.1 Defining Roles

Sowa [14] defines the role as follows: "Subtypes of the entity are of two kinds: natural types and role types, which are subtypes of natural types in some particular pattern of relationships. *PERSON*, for example, is a natural type, while *TEACHER* is

a subtype of *PERSON* in the role of teaching". Sowa proposes a simple test to decide whether a concept is a role: *r* is a role type if something can only be identified as type *r* only considering some other entity, action or state.

N. Guarino and C. Welty replace the condition formulated by Sowa for the condition of the so-called external ontological dependence:

> *Concept C1 is called externally dependent of concept C2 if for all examples of C1 there should be example C2 that is not a part or material of example C1* [10].

For example, *son* is externally dependent of parents because sons exist only within a family in relation to their parents. On the other hand, *car* is not externally dependent of an engine because it requires the existence of an engine that is a part of a car. Therefore, this condition specifies the definition of roles given by Sowa.

In addition, one more condition is introduced that, together with the condition of external dependence, provides better definition of the "role" concept.

> *Concept C is semantically rigid if any example of concept C remains example of C during the whole time of its existence.*

For example, an animal can cease to be a pup, but it is still a dog that is why *dog* and *animal* are semantically rigid, while a pup is not a rigid entity.

> *A concept is called a role if it is externally dependent and is not semantically rigid* [10].

3.2 Causes of Type-Role Confusion

Thus placement of roles as higher-level concepts for types is not subject to the most known principle of describing the taxonomic relationships and can arouse serious distortion in logical inferences. However this problem remains serious because it is "provoked" by many text sources.

For instance, the following fragment (http://www.giord.ru/0705211117391.php - translation from Russian):

> *the most widely used preservation substances are as follows: table salt, ethyl alcohol, acetic, sulfurous, sorbic and benzole acids and some their salts.*

may seem a good source of information for the description of kinds of preservation substances: table salt, ethyl alcohol, etc.

A definition of an electrolyte:

> *Electrolyte is the second-order conductor: the substance possessing ion conductivity. The electrolytes include:*
> *- melts of salts, oxides or hydroxides;*
> *- solutions of salts, acids or bases in polar solvents;*
> *and also solid electrolytes.*

may seem a sufficient ground, for instance, for defining the relationship that *chemical salt* is a kind of an *electrolyte*.

However, in such cases it should be remembered that *preservation substance* and *electrolyte* are roles of substances – an amount of substance becomes a preservation agent or electrolyte only in certain conditions. But a table salt and salt as a chemical compound are types of substances.

Defining the taxonomic relationship from a type to a role we communicate to a system the incorrect knowledge that any substance referred to the class of salts at any moment of its existence in any situation is an electrolyte, which is not so.

3.3 Description of Roles in Thesaurus RuThes and Ontology ONST

So, the question arises whether it is possible to describe the information received from the above fragments without too much complicating the model of knowledge representation? In Thesaurus RuThes and Ontology ONST we usually try to apply several methods.

First, if we assume that in our subject field most examples of a concept usually play a certain role, then we establish the taxonomic relationship but provide it with mark V, which means "possible by default".

Thus, for instance, we can establish such relationship between the concepts *SORBIC ACID* and *PRESERVATION SUBSTANCE*, if we assume that this is the main application of sorbic acid in our subject field (for example, food processing industry) and a probability to meet in the texts the sorbic acid in other applications, say, in organic synthesis, in our field is low:

SORBIC ACID
isa $_{V}$ *PRESERVATION SUBSTANCE*

However, it is not recommended to establish such a relationship between the concepts *TABLE SALT* and *PRESERVATION SUBSTANCE* because the main application of *table salt* is quite different. Even if we would establish such a relationship (for example, we could introduce one more mark for non-basic roles), then it should be taken into consideration that an automatic system can not determine from the context if it is possible to use this relationship or not.

Thus, in some cases we place the role concepts as higher-level concepts for type concepts, giving, however, such relationship a special mark. For each type one such relationship at maximum can be described indicating a default role for the type.

Using the description of the concept *ELECTROLYTE* we can demonstrate one more possibility for describing the relationships between the roles and types in the Thesaurus RuThes and Ontology ONST.

We try to introduce an additional concept for the situation when *salt* is in the role of *electrolyte*. If this situation is essential for the given domain, then our attempt is usually supported by the language of a subject field – for such a concept there is always one or more commonly used language expressions. And in our case such expression as *electrolyte salt* exists and is actively used.

Therefore, we can introduce concept *ELECTROLYTE SALT* and establish the following relationships:

ELECTROLYTE SALT
isa *CHEMICAL SALT*
isa *ELECTROLYTE*

In this way we correctly reflect the knowledge we received from the read definition.

If we apply such an approach to relationships between the concepts *EMPLOYER, PERSON* and *ORGANIZATION*, we will have to introduce two additional concepts, such as for employer as a natural person *PERSONAL EMPLOYER* and for an employer as a legal entity – *CORPORATE EMPLOYER* (fig.2).

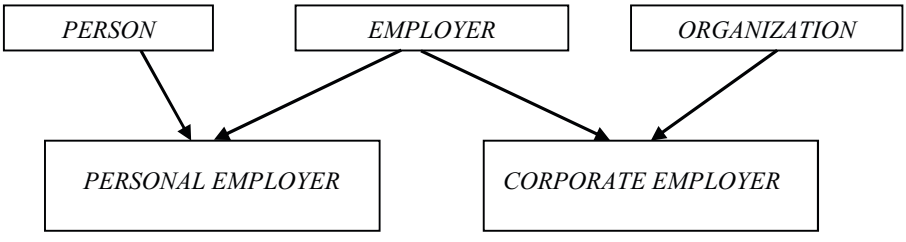


Fig. 2. Description of relationships between concepts *EMPLOYER, PERSON, ORGANIZATION*

PERSONAL EMPLOYER
isa *EMPLOYER*
isa *PERSON*

CORPORATE EMPLOYER
isa *EMPLOYER*
isa *ORGANIZATION*

As introduction of additional concepts may complicate significantly the description of concepts in a resource, such a method is applied only if such additional concepts are really found in the subject field as is in the case with the concept *ELECTROLYTE SALT*. It's worth mentioning that the introduced additional concepts *PERSONAL EMPLOYER* and *CORPORATE EMPLOYER* also have real usage in the law domain because the law regulates differently the relationships of different types of employers with employees.

4 Confusion of Taxonomic and Instance-Class Relationships

The contemporary ontological modeling [3, 12, 15] distinguishes quite clearly the relationships of instantiation (instance-class relations) from the taxonomic relationships. Instantiation relationships link individual entities, for instance, such as a particular city – *Moscow* and classes of entities, such as *CITY*. Unlike the taxonomic relationships the relationship instance-class is not a transitive relationship.

Many guidebooks state that instances are the most concrete conceptual units represented in a database. Thus, the work [5] provides such an example: if we have to describe only the selected combinations of wine and food, then we will be interested only in concrete material bottles of wine, thus, such terms as *Sterling Vineyards*

Merlot will be the most concrete applicable conceptual units. Consequently, *Sterling Vineyards Merlot* will be an instance in a knowledge base and the instantiation relationship should be established between this wine and a class of wines.

The complexity that leads to confusion of these two kinds of relationships consists in the fact that the instantiation relationship may be met at any hierarchical level of a conceptual system and not only at the lowest levels.

Thus, the concept *SPANIEL* is linked by the taxonomic relationship with the concept *DOG* and by the instantiation relationship with the concept *DOG BREED*; the concept *SCHOOL TEACHER* is linked by the taxonomic relationship with the concept *EDUCATIONAL WORKER* and by the instantiation relationship with the concept *PROFESSION*. In such cases it is not always easy to distinguish such relationships.

For distinguishing between the taxonomic and the instantiation relationships we may use the principle of identity (see section 2) asserting that a generic concept and a specific concept should have the same identity criteria.

If we analyze the identity criterion, for instance, for the concepts *SPANIEL* and *DOG BREED*, we will see that the identity criteria for spaniels and animal breeds are different. The dog breeds are identified with their position in a certain dog classification. On the other hand, the instances of spaniels may, in the simplest cases, be identified via location of their bodies in space/time – two spaniels are different if they locate in one and the same time in different places. Therefore the concept *DOG BREED* cannot be a generic concept for the concept *SPANIEL*. A spaniel is not a subclass of dog breeds, but an instance.

Likewise a particular teacher is identified by its physical location, while professions – by some set of characteristics: an educational level, work experience, necessary skills. Hence, the concept *SCHOOL TEACHER* is an instance of the concept *PROFESSION*, but not a subclass.

5 Confusion of Taxonomic Relationships and Part-Whole Relationships

The example of an erroneous chain of relationships given in section 2:

> *RIVER* – isa – *WATER BODY* – isa – *WATER OBJECT* – isa – *WATER* – isa – *SUBSTANCE,*

also corresponds to one of the most widespread types of problems arising at the description of the taxonomic relationships.

The essence of the problem consists in the fact that some entity has an essential part that takes a great share of the volume in this entity and then it seems that it is possible to transfer the taxonomic relationships of this part to the whole entity.

This mistake is not revealed by diagnostic language tests. Thus, expressions “river is water”, “river and other water” sound quite good.

In addition, a mistake may be “provoked” by definitions in dictionaries: “River is a large area of water that flows towards the sea” [10]. So, from such a definition we can make a conclusion that a river is water.

The Water Code of the Russian Federation (RF Federal Law No. 167-FZ of 16.11.1995) provides the following definition of the concept WATER OBJECT:

> *Water object is concentration of water on the land surface repeating its relief or inside the earth that is characterized by borders, volume and features of a water regime (Article 1).*

From such a definition it can be concluded that a water object is water.

In such cases the comparison of the top-level concepts corresponding to supposed specific and generic concepts may be helpful – in case of an erroneous relation such top-level concepts may be different and this fact contradicts to properties of taxonomic relationships.

In addition, an incorrect relationship is revealed by the identity analysis of the specific and generic concept: destruction of a river does not result in destruction of water – water simply runs to some other place.

Therefore, the more accurate description of relationships between concepts *RIVER* and *WATER* may be as follows:

RIVER
part *RIVER WATER*

RIVER WATER
isa *WATER*
whole *RIVER*

One more example of the same problem is the relationship between the concepts *COMPANY* and a *GROUP OF PEOPLE* [12].

6 Confusion of Taxonomic and Origin Relationships

One more kind of confusion of relationships that was already mentioned in section 2 concerns the incorrect description of the origin relationship as the taxonomic relationship as for concepts *RAISIN* and *GRAPE*. Similar to the previous cases such a mistake is often based on dictionary definitions. Thus, in WordNet2.1 amber is defined as "*a hard yellowish to brownish translucent fossil resin; used for jewelry*". But it is not correct to describe that concept *AMBER* is a specific kind of the concept *RESIN*, *amber* originates from *resin*.

Such a mistake may be identified by the analysis of properties and relationships of the specific and generic concepts. A specific concept obtained as a result of confusion with the origin relationship does not inherit many properties and relationships of the generic concept and does not inherit likewise relations to top-level concepts (see Section 2).

7 Taxonomic Relationships and Clustering of Word Senses

Many developers of ontological resources for natural language processing as well as developers of ontologies for other applications face the problem of polysemy of words (lexical polysemy).

In the first case the developers understand that introduction of additional senses in a computational vocabulary will create an additional burden for the processing system to choose among different senses [16].

Developers of conceptual resources for non-linguistic applications face the polysemy problem in the course of domain analysis when it is necessary to reveal the required conceptual system of the domain. And this procedure may become very difficult due to lexical polysemy, for example, in such cases when the senses of a term are related closely to each other.

For instance, in Russian in trading domain there are two senses of the word "*prodavec*":

> *1. somebody who is employed to assist and sell goods to customers in a retail store (corresponds to English salesclerk)*
> *2. somebody who is selling: a person, store, or company that offers something for sale.(corresponds to English seller.1)*

Such senses (both ones sell something) seem so close that there is a desire to cluster these two senses to one conceptual unit.

So, the too detailed set of senses of words is often mentioned as one of the serious causes complicating utilization of WordNet [11]. Many authors studied different automatic methods to cluster the most related senses of WordNet [17].

At the same time it should be noted that the combination of even close related senses leads to a situation when a respective conceptual unit will have two and more generic concepts. And such taxonomic relationships may be relevant to one textual context and non-relevant to some other, e.g. *prodavec* in the first sense may have such higher-level concept as *TRADE WORKER,* which will be incorrectly to apply to the usage of *prodavec* as *selling company*.

Therefore, if the described relationships are to be used for logical inference, it will be necessary to define, first of all, whether this relationship is applicable to a given context. This means that the problem of selection of a sense of an ambiguous word will be simply moved to other stages of text processing. Besides the main principle of the taxonomic relationships description is violated.

For close-related senses distinguished by the taxonomic relationships it will be more appropriate to introduce two separate conceptual units and to establish relationships between them [18].

8 Conclusion

We described the typical problems faced in the course of establishing the taxonomic relationships in ontological resources. We have demonstrated that the linguistic tests and definitions from dictionaries and encyclopedia should be applied with great caution.

Establishing the taxonomic relationships developers should use a set of formal criteria.

Implementation of a real applied task may urge to violate the described formal criteria. But it is important that such violation of criteria was a conscious choice and developers of resources have to know about consequences of their decisions.

References

1. Cruse, D.: Lexical Semantics. University Press, Cambridge (1986)
2. Guarino, N.: Some Ontological Principles for Designing Upper Level Lexical Resources. In: Proceedings of First International Conference on Language Resources and Evaluation, Granada, Spain (1998)
3. Noy, N.F., McGuinness, D.: Ontology Development 101: A Guide to Creating Your First Ontology. Technical Report KSL-01-05, Stanford Knowledge Systems Laboratory (2001)
4. Z39.19 – Guidelines for the Construction, Format and Management of Monolingual Thesauri. NISO (1993)
5. Gomez-Perez, A., Fernandez-Lopez, M., Corcho, O.: OntoWeb. Technical Roadmap. D.1.1.2. IST project IST-2000-29243 (2001), http://www.aifb.uni-karlsruhe.de/WBS/ysu/publications/OntoWeb_Del_1-1-2.pdf
6. Nirenburg, S., Wilks, Y.: What's in a symbol: Ontology, representation, and language. J. of Experimental and Theoretical Artificial Intelligence 13(1), 9–23 (2001)
7. Loukachevitch, N., Dobrov, B.V.: Development and Use of Thesaurus of Russian Language RuThes. In: Proceedings of Workshop on WordNet Structures and Standardisation, and How These Affect WordNet Applications and Evaluation, Gran Canaria, Spain, pp. 65–70 (2002)
8. Dobrov, B., Loukachevitch, N.: Development of Linguistic Ontology on Natural Sciences and Technology. In: Proceedings of Linguistic Resources and Evaluation Conference (2006)
9. Will, L.: Thesaurus consultancy. In: Roe, S.K., Thomas, A.R. (eds.) The Thesaurus: Review, Renaissance and Revision. Haworth, New York (2004)
10. Macmillan English Dictionary for Advanced Learners. Publishing Plc, Blumsbury (2002)
11. Miller, G.: Nouns in WordNet. In: Fellbaum, C. (ed.) WordNet – An Electronic Lexical Database, pp. 23–47. The MIT Press, Cambridge (1998)
12. Guarino, N., Welty, C.: Evaluating ontological decisions with ONTOCLEAN. Communications of the ACM 45(2), 61–65 (2002)
13. Steinmann, F.: The representation of roles in object-oriented and conceptual modeling. Data and Knowledge engineering 35(1), 83–106 (2000)
14. Sowa, J.: Using a Lexicon of Canonical Graphs in a semantic interpreter. In: Evens, M. (ed.) Relational Models of Lexicon, pp. 113–137. University press, Cambridge (1988)
15. Cyc Ontology Guide: Introduction, http://www.cyc.com/cyc-2-1/intro-public.html
16. Nirenburg, S., Raskin, V.: Ontological Semantics. MIT Press, Cambridge (2004)
17. Gonzalo, J., Chugur, I., Verdejo, F.: Sense clustering for information retrieval: evidence from Semcor and the EWN Interlingual Index. In: Proceedings of the ACL Workshop on Word Senses and Multilinguality (2000)
18. Gonzalo, J.: Sense Proximity versus Sense Relations. In: Proceedings of International Wordnet Conference, pp. 5–6 (2004)

Problems in Constructing an Empirical Theory of Data Mining

Nikolay G. Zagoruiko

Sobolev Institute of Mathematics of the Siberian Branch
of the Russian Academy of Sciences,
pr. Koptyug 4, Novosibirsk, 630090, Russia
zag@math.nsc.ru

Abstract. The paper describes the structure of empirical theories and analyzes the nature of problems inherent in data mining. A formal description an empirical theory of data mining is given. A general approach to the construction of data mining methods based on the function of rival similarity (FRiS-function) is presented. Application of this function allows the construction of a new class of data mining methods and strengthens empirical theory.

Keywords: Empirical theories, data mining, discovery of regularities, function of rival similarity, potential refutability.

1 Introduction

The main distinctive property of systems which possess intelligence is their skill of correct prediction. Such prediction becomes possible only if, in a stream of events, it is possible to find natural connections between observable situations. Consequently we have a system of approximately true knowledge of the world around, surrounding by which are certain empirical hypotheses or theories [1–3]. The word "empirical" means that such theories and hypotheses can be proved to be true or not only by results of supervision over events in the real world. Theories should be constructed concerning Dimitri Mendeleyev's criterion that the purpose of science is truth and benefit. The truth of the empirical theory consists to a large extent of confirmation of laws resulting from abundant supervision. The benefit of the theory consists of available proved recommendations.

2 What Is an Empirical Theory?

In formal representation the empirical theory or the empirical hypothesis, h, can be represented as: $h =< W, O, V, T >$. Here the symbol W is a set of objects or the phenomena relevant for the theory h. For example, "all material bodies" or "all methods of pattern recognition".

The symbol O is the instruction denotes the set of instructions describing the intended supervisions concerned with the considered theory, and V denotes

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 243–255, 2011.

the language in which the results of the supervisions should be formulated. The instruction should answer the question: is the given supervision performed according to this instruction or not? Any such supervision should suppose the description of results in the form of the protocol pr^v in the dictionary of language V.

Symbol T designates a "test" algorithm which works with any protocol in language V and has one of two values: $T(pr^v) = 1$ if the protocol is to be coordinated with a hypothesis h, or $T(pr^v) = 0$, otherwise. For isomorphic protocols (the protocols with identical empirical contents), algorithm T yields identical values.

The empirical sense of the theory h is defined by the following agreement: world W is that for which, if we observe accordance with O, we shall never receive a protocol pr^v on which the test algorithm will assign value $T(pr^v) = 0$.

Empirical theories describe potentially refutable hypotheses about the device of the studied world. If h is not denied by the protocol pr^v it is accepted that the theory h will agree with the given supervision, and the protocol pr^v confirms the theory. The theory h can never be proved once and for all, but it can be denied by just one experiment.

The empirical theories concerning objects and the phenomena of some applied areas, can describe natural connections between different subsets of their characteristics. So, developed theory H of a complex empirical system can include in the structure some empirical theories of type h. For example, in the field of data mining (DM) it is possible to formulate the theory h for methods of decision a tasks of each type of: the theory h_s for methods of automatic classification, the theory h_x for methods of a feature selection, etc. Alongside these, it is possible to construct theory H for describing some general properties of DM methods for tasks of a any type. The term "an empirical hypothesis" we shall to use for theories of a solving tasks of one type, and the term "the empirical theory" - for theory describing the general properties of methods a solving any DM tasks.

Apart from the empirical contents of theories and hypotheses it is possible to note other properties, such as potential refutability Q, degree of confirmation P, depth of explanation E, simplicity S and beauty B of the formulation [1]. The regularities is a empirical hypothesis h, accompanied by its characteristics:

$$R =< h, Q, P, E, S, B > .$$

Development of the theory is connected with strengthening hypotheses, with improvement of these characteristics. The most important is the potential refutability Q. Improvement Q is connected with transition, for example, from incontestable and empty statements of the type "In this world all is possible" to the very risky, but not denied yet , statement "Force is equal to the weight increased by acceleration". The more imagined protocols can deny a hypothesis, the greater the value of Q and the more for practice it is useful . Our trust to a hypothesis also grows with increase in quantity P of the confirming experiments. Answers to questions of how and why give a deeper explanation of

E hypotheses. In the history a science there are many examples showing utility simple (S) and beautiful (B) of formulations of hypotheses.

From the above it is clear that for construction of the empirical theory of some applied areas it is necessary to define the objects W of this area, what their properties O means and to write it in language V. It is also necessary to describe algorithm T by means of which we shall define what is possible or impossible in this area. We shall try to apply this approach to a domain which is connected with data mining, i.e. with methods of automatic discovery of empirical regularities and their use for predictions.

3 State-of-the-Art DM Problem

The problem of discovering empirical regularities is central to Artificial Intelligence. The mathematical methods used for discovering empirical regularities (knowledge) is named as Intellectual Data Analysis or Data Mining. These methods are included as objects W in our study also. Among DM tasks, which are included in the classification [1], the following are the most popular:

1. **A task of type S:** creation of classification structure of the set of M objects. Using algorithms of classification the set of M is divided on k classes (clusters) by similarity of characteristics of objects. . Received classifications can have single-level or multilevel hierarchical structure. Borders between classes can have a simple linearly form or figures of any complexity.

The empirical hypothesis, which connect with methods of automatic classification h_s can be so: $h_s =< W, O, V, T >$, where

W – all possible methods w_1, w_2, $\ldots, w_j$, $\ldots, w_J$ of partition of M on k clusters S_1, S_2, $\ldots, S_k$. Parametr k can be in the limits from k_{min} up to k_{max};

O – ways of estimating and recording in language V characteristics of received variants of splitting. Among characteristics of clustering are such "geometrical" characteristics as a measure of similarity of objects in one cluster, remote clusters from each other, etc. It is also necessary to pay attention to "inductive" properties of received classification: if one knows some properties of cluster objects then it is possible to define other properties of these objects. This requirement is usual for algorithms of so-called "natural" classification [4, 5];

T – the test algorithm which receives the protocol pr^v estimates the quality F described in the protocol of classification and returns the decision 1 if quality exceeds set threshold F^*, and 0 in other cases.

Such a hypothesis allows us to divide all possible methods of clustering on admissible (W_+) and inadmissible (W_-). It is possible to arrange algorithm T so that it compared protocols pr^v and gave out 1 to only one of them, the best algorithm of classification. Potential refutability Q such a hypothesis h_g, that a certain algorithm w_j is better than others, is proportional to the quantity of algorithms-competitors. The degree of confirmation P this hypothesis above if the more tests was won with algorithm w_j. Characteristic E will above if explanation how algorithm w_j works is more deep. It will be useful if the algorithm can be described simply.

The contents of all other tasks of DM and corresponding hypotheses about methods for their decision can be described similarly.We shall be limited to the description the contents of tasks of different types.

2. **A task of type X:** formation of system X of the informative description of objects of set M preliminarily divided on k of classes. The system of N primary attributes is set by experts. At the decision of this task a method of "filtering" from N attributes produces the most informative subset from $n < N$ attributes. With the method of "selection" N primary attributes will be transformed to a more informative system from n secondary attributes [6].

For an estimation of the informativeness of attributes a criteria of two kinds are used: indirect and wrapping. The indirect estimation includes an entropy of density of patterns in separate parts of space of attributes, complexity of structure of logic decision rules, distance between patterns, divided by the sum of their dispersions (Fisher's criterion), etc. It is considered that direct methods (One-Leave-Out (OLO), Cross-Validation (CV), etc.) are more time-consuming but also more reliable criteria of informativeness. The percentage correctly recognized test objects withdrawn from the training sample is accepted as expected reliability of the future recognition of the real control sample.

3. **A task of type D:** there is a set of training objects in which each of M objects is carried to one of k classes, and decision rule D for reliability recognition of a new objects are under construction. As in the previous case, indirect and direct methods of an estimation of quality of the constructed decision functions are used.

4. **A task of type Z:** filling of blanks (the analysis of incomplete data or "inserting") and detection of errors in data tables (cleaning of data) [1]. Natural connections found between different parts of the table of data are used for a prediction of the most plausible value of the missed or distorted element. The expected error in filling a blank is estimated by a direct method – by criterion of a minimum of mistakes of a prediction of known elements of the table. A task of this type is also, if necessary, applied to the detection of deviations from norm (in particular, fraud detection). Thus it is necessary to generate the description of the object or process "in norm" and to learn the program to find out in good time deviations from norm.

5. **A task of type P:** forecasting of dynamic processes. According to this task, describing observable dynamics of development of process, it is necessary to find natural connections of the past with the future and then to predict characteristics of process during the set future moments of time. During the learning the criterion of quality in the form a minimum of mistakes of forecasting in the retrospective analysis is applied.

6. **A task of type A:** detection of associations. It is necessary to find steady connections between the values of one subset of characteristics X_1 and the other subset of characteristics X_2. Then, from known values X_1 it is possible to predict values of characteristics X_2. Here again an expected mistake of the future decisions is usually evidenced by the results of the retrospective analysis.

Apart from these tasks of the basic types, there are tasks of the combined type: for example, a task of taxonomy with a simultaneous choice of the most informative subspace of attributes (**a task of type SX**). Or, even more complex, the challenge of simultaneous construction of classification, a choice of attributes and constructions of a decision rule (**a task of type SDX**). The situation becomes complicated if data are described by polytypic attributes. Many real tasks are statistically misused: the quantity of attributes happens to be the same as, and can even exceed, the quantity of objects

The great variety and complexity of DM tasks and their high urgency have meant that for the last thirty to forty years. Many algorithms have been developed for the decision of each of them. Attempts to constructing the ontology of DM tasks and methods [7] has shown the lack of a uniform approach to the decision of not only the combined tasks but each separate task.

4 Reference Points for Further Development

What should guide us in the search for a uniform approach to the performance of different DM tasks? We note that all DM methods to some extent simulate the human ability to systematize the world around, to form classifications, to choose the most important characteristics of classes, to define the belonging of new objects to this or that class. It is clear, we must proceed by way of rapprochement with formal models which have properties like the human mechanism of orientation in the world around. On which features of this mechanism we should concentrate?

1. The natural mechanism is based on a universal approach allowing us easily to combine results of decisions concerning different tasks in a uniform consistent chain of decisions. So, reasonable natural classifications (a task of type S) are based on the use of a small number of characteristics (a task of type X) from which it is easy and quite possible to recognize an object belonging to the class (a task of type D).

From here, the requirement arises for a coordination of methods concerning different tasks. It is necessary to note, that the requirement for a **coordination** of results can be met by some of the existing methods. If the combined tasks are solved by means of a uniform approach, for example, the algebraic approach [8] or logic trees [9], results will be coordinated among themselves. But it is impossible to suppose that, for example, the classes received by a the method k-means were recognized by means of logic decision rules (trees) in which planes are used, perpendicular to axes of coordinates.

2.If laws of distribution of classes are known, it is possible to provide compatibility, applying the methods which focus on these types of distribution. Unfortunately, little is known about the character of distributions, but human solve such tasks. The same relates to such features of real tasks as the character of dependencies between attributes and the ratio between quantity of objects and attributes. The human mechanism is prepared for all these displays of the real world. The requirement for of DM methods to be **invariance** to a kind of law

of distribution, the character of dependencies between attributes and statistical conditionality of a task, is a natural corollary.

3. Clearly, that at simplification of a task (for example, distributions are normal, the quantity of attributes is not enough, a lot of training objects), methods should give decision which is nearer to optimum. From this the requirement for **potential optimality** of DM methods follows.

4. It is well-known that DM results strongly depend on whether training set is representative and whether objects are independent. But, unfortunately, we cannot know all the objects of a class are represented in the training sample and whether it is sufficiently full. The only thing which is necessary, is to copy from the human mechanism its constant readiness to adapt models of classes with the occurrence of new objects. The person includes a new object in a class if the presence a this object will be maximal compatible with the model of the given class.

5. All methods of data analysis use this or that measure of affinity or similarity. Human abilities to estimate remote similarity or to find thin distinctions differ very greatly in terms of efficiency. The closer the formal measure of similarity gets to **simulating these abilities**, the more successfully DM algorithms DM will work. Research into different measures of similarity has led us to a conclusion about the expediency of using a measure which we have named the Function of Rival Similarity or FRiS-function. This function, it seems to us, nicely simulates the human mechanism of estimating similarity. On the basis of FRiS-function it is possible to construct effective algorithms of the decision of all primary tasks of pattern recognition [10–12]. These algorithms meet all the listed requirements. They possess properties of compatibility and harmony, potential optimality and invariance in proportion with the number of objects M and numbers of attributes N, character of distributions and kinds of dependencies between attributes. We will explain that is means a FRiS-function.

5 Function of Rival Similarity

Let's try to formulate the main properties which the function of similarity F should possess.

1. Dozens of various measures of similarity [13] are described in the literature. As a rule, in these measures similarity of control object z to standards has a absolute category and depends on distances to these standards only. But it is easy to demonstrate that the human perception of similarity has a relative nature. To answer questions like "Is it similar or not?" one should know the answer to the question "In comparison with what?". Function F should therefore measure relative value of similarity depending on the peculiarities of a competitive environment.

Some existing algorithms of recognition operate according to this principle. For example, under the "k nearest neighbours" (kNN) recognition rule,a new object z is classified as an object of the i-th pattern S_i if the distance $r_{z,i}$ to this pattern is not only small but is less than the distances $r_{z,j}$ to any other rival patterns S_j.

2. A person can estimate a measure of similarity not only in the scale of order ("it is more similar to S_i than to S_j"), but also give a value of similarity in the absolute scale. So function $F_{z,i|j}$ of similarity of object z to the standard of i-th pattern S_i could range from -1 to 1 and should amount to extreme values in two cases: 1 if object z coincides with standard S_i, and -1 if object z coincides with the standard of pattern-competitor S_j. In the case of identical similarity of the object to both standards, function $F_{z,i/j} = 0$.

The Function of Rival Similarity possesses these properties

$$F_{z,i|j} = (r_{z,j} - r_{z,i})/(r_{z,j} + r_{z,i}).$$

FRiS-function has been useful in constructing a methods for the decision of recognition tasks of all types.

6 Construction the Decision Rule (Algorithm FRiS-Stolp)

The process of training the data set before recognition with use of the FRiS-function consists of choosing a subset of a subset of samples (whose elements are called stolps), which will be used for recognition of control objects. The algorithm for choosing a stolps should be independent of the type of probability distributions of the patterns. The decision it finds out should be adaptable for any situation. In case of unimodal distributions, stolps should settle down in the centers of gravity of patterns. If distributions are polymodal and patterns linearly not divisible, stolps should stand in the centers of local compactness. While distributions get more complex the number of stolps k should increase.

Algorithm FRiS-Stolp possesses these properties. It is intended to select the minimal number of stolps which protect all training samples from incorrect recognition. The algorithm is in detail described in [14]. Here we shall briefly describe the basic idea of this algorithm in case of two patterns S_i and S_j.

All objects of training sample of the pattern S_i by turns play a role of a stolp. From other objects of this pattern two distances are measured: up to this stolp (r_i) and up to the nearest object of any other pattern (r_j). On these distances the measure of similarity F of object with the stolp is calculated. If F is greater than the threshold value of similarity F^* it is considered, that this object is protected by the stolp. The object which in a role of a stolp protects the greatest quantity of objects of the pattern S_i is defined. It becomes the first stolp of this pattern.For objects which remained not protected, this procedure repeats. Then the same competition is spent among objects of all other patterns. The list of stolps for all patterns as a result turns out. The decision rule using these stolps for recognition of control object z consists in the following. Distances $r_{z,i}$ and $r_{z,j}$ from z up to the two nearest stolps belonging to different patterns are estimated. This object is classified as an object of pattern S_i, if the value of similarity $F_{z,i|j}$ to its stolp is maximum.

7 Selection of Informative Attributes (Algorithm FRiS-GRAD)

Before describing an algorithm for the choice of informative attributes, we shall consider methods for an estimation of informativeness.

If distributions of patterns are unknown, the informativeness of attributes is usually estimated by the method OLO or CV. We assert that the average value of FRiS-function is a more accurate estimation of informativeness. If, for example, objects of two patterns are presented by two linearly-divided classes of objects, the quantity of correctly-recognized objects (here 100%) doesn't depend on the distance between groups. But the average value of the function of rival similarity (F_s) depends on how close these pairwise disjoint classes are to the dividing border.

This hypothesis was checked by the following experiment with initial data consisting of 200 objects of two patterns (each consisting of 100 objects) in a 100-dimensional vector space over the real numbers. Attributes were generated so that they possessed different informativeness. As a result about 30 attributes appeared to some extent informative, and other attributes were generated by the random-number generator and were obviously not informative. In addition, the given tables were distorted by noise of different intensity (from 0.05 up to 0.3). For the training set, 35 objects of each pattern had been selected; for the control set, the other 130 objects had been chosen. At each noise level by means of algorithm AdDel [15] the two best n-dimension subsystems of attributes (n from 1 up to 22) got out – one on a minimum of errors of recognition of training sample by method OLO (criterion U) – and the second on the maximal value of the average value of FRiS-function. Each of the two subsystems was estimated by two sizes: the share of correctly-recognized objects of training sample by method OLO and the share of correctly-recognized control objects. Results are presented in Fig. 1. Here, thin lines show the reliability of the recognition of the training sample, and thick lines the reliability of the recognition of the control sample. It is obvious that criterion F_s allows us to estimate the informativeness of attributes more objectively. Reliability of the recognition of the training sample predicts reliability of recognition of the control sample more precisely. Criterion U gives the overestimated level of quality of attributes and conjures illusions which do not prove to be true in the control sample.

To solve the problem of the selection a subset of most informative attributes from a large initial set, we can use an arbitrary directional searching algorithm, e.g., the AdDel algorithm [15] or GRAD algorithm [16]. These algorithms automatically determine both the content and the best quantity of characteristics. Here, we notice that these algorithms involve a new criteria of informativeness based on the use FRiS-functions.

8 Construction of Classifications (Algorithm FRiS-Tax)

At the decision of a task of type S, automatic classification of objects in the form of a hierarchy of classes, or the list of classes of one hierarchical level, can be

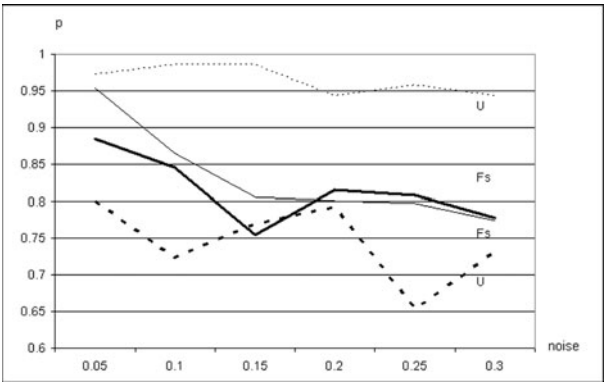


Fig. 1. Results of training and recognition by criteria U and F_s at different levels of noise. Thin lines – the training; thick lines – the control.

done by means of the algorithm FRiS-Tax [14]. Its work consists of two stages. At the first stage the algorithm FRiS-Cluster selects the objects which are centers of local clots of objects. Such objects become standards (stolps) of clusters. At the second stage, by means of algorithm FRiS-Class, there is a procedure of integration of clusters in classes (taxons) by association of some further clusters in one class. It allows us to create classes of any form, not necessarily linearly-divisible.

If we find the average value of the function of rival similarity F_s of all objects with stolps of clusters of this size, we can characterize the quality of clustering. It has been shown that at change of cluster quantity k local maximums of function $F_s = f(k)$ take place at such values k, which experts consider as the most preferable. It allows **automation of a choice of the best quantity of clusters k.**

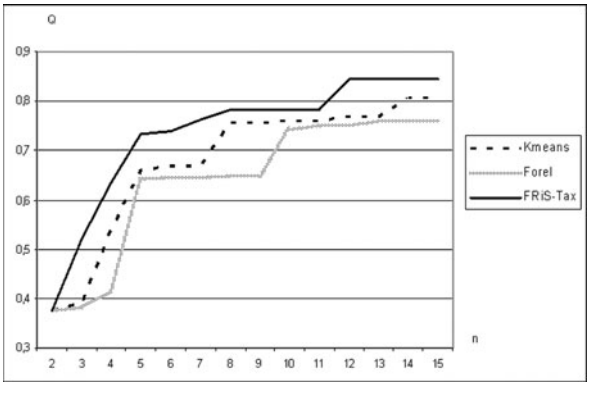


Fig. 2. Comparison of the quality of three algorithms of classification

Comparison of algorithm FRiS-Tax was done with other algorithms operating the concept of the center of cluster – with algorithm k-means [17, 18] and Forel [1]. Results have shown that it exceeds them in the quality of received decisions (see Fig. 2).

9 Application of FRiS-Function to the Decision of Other DM tasks

The use of FRiS-function allows to us to realize easily algorithms for the decision of tasks of combined type [14]. So, the task of type DS – simultaneous construction of classification (S) and a decision rule (D) – is solved directly during taxonomy method FRiS-Tax: taxons are described by standards (stolps) of clusters through which the recognition of new objects is conducted. We realize algorithms of type DX (construction of a decision rule in the most informative subspace of attributes) and SX (taxonomy in the most informative subspace). The last case actually coincides with a complex algorithm of combined type SDX. Thus attributes get out by means of algorithm FRiS-GRAD [16], and the taxonomy in every subspace is done by algorithm FRiS-Tax which in passing gives out decision rules in the form of system of stolps.

Our experience shows, that FRiS-function can serve as a useful element of DM algorithms intended for the decision of tasks a various types, including filling gaps (a task of type Z), forecasting (a task of type P) and search of associations (a task of type A).

10 Strengthening of the Empirical DM Theory

The empirical theory describing the general properties of modern data mining methods can be presented thus: $H_1 =< W_1, O_1, V_1, T_1 >$, where

W_1 – set of all possible methods of the decision of DM tasks of basic and combined types.

O_1 – the list of characteristics for each method.

The concrete DM methods can differ from each other in the following characteristics:

– types of tasks on which the method is focused,
– requirements for statistical conditionality of data,
– orientation to kinds of laws of distribution,
– types of measuring scales with which the method works,
– the metrics of space of attributes,
– criterion of success of the decision of a task,
– presence of positive experience of application of a method,
– presence of the accompanying information (explanations, instructions), etc.

V_1 – language for recording values of measured this characteristics of methods.

T_1 – test algorithm which divides methods into admissible and inadmissible with respect to the values of this characteristics.

Research on the functions of rival similarity described above shows the suitability of the uniform approach to the decision of different tasks that provides compatibility of methods at the decision of tasks of the combined type. Application of FRiS-function for an estimation of quality of offered variants of decisions also produces increased reliability of accepted decisions. These results allow us to extend the list of characteristics which the admissible DM method should possess, and to make changes to test algorithm T which increase its filtering properties.

Available strengthening of general empirical DM theory is shown in the following. Theory H_2 differs from H_1 in the list of characteristics where the following items are added:(see section 4):

– a coordination of results at the decision of tasks of the combined type,
– invariance – a method for one sort of law of distribution,
– invariance – a method for statistical conditionality (to a ratio of number of objects and attributes),
– a harmony of a method,
– a potential optimality of a method.

To the list of values should also be added the criterion of success of the decision of a task and the indirect criterion based on FRiS-function.

If the test algorithm of theory H_2 demands from a DM algorithms the presence of all these properties, it will consider unacceptable those methods which demand knowledge of the type of distributions, distributions, which are critical to the ratio of the number of objects and attributes, and those methods which do not possess properties of a potential optimality and a harmony.

It is possible that the test algorithm will estimate not all additional characteristics of a method, but some part only with regard to the strengthening of theory H_1 to theory H_2. For example, it will not demand from a method of invariancy laws of distribution, or suppose the use of a traditional method for an estimation of informativeness by the number of errors of recognition of objects of a training sample in cross-validation mode. Such variants of theories will take up an intermediate position between theories H_1 and H_2.

Potential refutability Q_2 offered theory H_2 it is essential more strong of Q_1 modern data mining theory. Further strengthening of these theory characteristics with respect to confirmation P_2 and explanation E_2 will be continued to yield a high efficiency of the new approach.

11 Conclusion

Knowledge of modern DM methods allows us to systematize them in the form of ontology and to show the basic structural elements of one version of future empirical DM theory.

Continuation of researches is required into methods of inductive conclusion, specification the characteristics of DM methods, development a ways of measuring their values and development of variants of test algorithms which would allow us to choose the admissible methods applicable to a specific target.

For development of existing DM methods the function of rival similarities (FRiS-function) can be used as a universal basis for algorithms solving all basic and combined DM tasks with any degree of statistical conditionality and to any character of distribution of analyzed objects in space of attributes. These facts provide the basis for strengthening potential refutability Q of the existing empirical DM theory.

Acknowledgments. Work has been executed with the support of the Russian Federal Property Fund (grants 05-01-00241 and 08-01-00040). The author expresses sincere gratitude to K.F.Samochvalov, who has been greatly influential in developing research on methods of empirical prediction, and to I.A. Borisova, O.A. Kutnenko and V.V. Dyubanov for active discussion of the problem presented here and for execution of numerous machine experiments.

References

1. Zagoruiko, N.G.: Applied Methods of Data and Knowledge Analysis. Institute of Mathematics SD RAS, Novosibirsk (1999) (in Russian)
2. Samochvalov, K.F.: On Theory of Empirical Predictions. Computer Systems 55, 3–35 (1973) (in Russian)
3. Zagoruiko, N.G., Samochvalov, K.F., Sviridenko, D.I.: Logic of Empirical Researches. Novosibirsk State University, Novosibirsk (1978) (in Russian)
4. Vityaev, E.E.: Algorithm of Natural Classification. Computer Systems 99, 44–50 (1983) (in Russian)
5. Borisova, I., Zagoruiko, N.: Princips of natural classification. In: 7-th International Conference on Pattern Recognition and Image Analysis: New Information Technologies (PRIA-7-2004), pp. 28–31. St. Petersburg (2004)
6. Ivakhnenko, A.G.: Polynomial theory of complex systems. IEEE Transactions on Systems, Man and Cybernetics. SMC 1(1), 364–378 (1971)
7. Zagoruiko, N.G., Gulyaevsky, S.E., Kovalerchuk, B.Y.: Ontology of Subject Domain "Data Mining". Pattern Recognition and Image Analysis 17, 349–356 (2007)
8. Zhuravlev, J.I.: Selected scientific works. URSS, Moscow (1988)
9. Lbov, G.S., Startceva, Y.G.: Logic decision Function and problems of Statistical Stability of Decisions. Institute of Mathematics SD RAS, Novosibirsk (1999) (in Russian)
10. Zagoruiko, N.G., Borisova, I.A., Dyubanov, V.V., Kutnenko, O.A.: Methods of Recognition Based on the Function of Rival Similarity. Pattern Recognition and Image Analysis 18, 1–6 (2008)
11. Borisova, I.A., Zagoruiko, N.G., Kutnenko, O.A.: The Criterion of Informativeness and Suitability Subset of Attributes Based on Function of Similarity. Zavodskaja Laboratoruja 74, 68–75 (2008) (in Russian)
12. Zagoruiko, N., Borisova, I., Dyubanov, V., Kutnenko, O.: Function of Rival Similarity in Pattern Recognition. In: 8th International Conference on Pattern Recognition and Image Analysis: New Information Technologies (PRIA-8-2007), vol. 2, pp. 63–66. The Russian Federation, Yoshkar (2007)
13. Voronin, J.A.: Beginning of Theory of Similarity. Computer Centre SD RAS, Novosibirsk (1989) (in Russian)

14. Borisova, I.A., Dyubanov, V.V., Zagorujko, N.G., Kutnenko, O.A.: Use of FRiS-function for taxonomy, attributes selection and decision rules construction. In: Wolff, K.E., et al. (eds.) KONT/KPP 2007. LNCS (LNAI), vol. 6581, pp. 256–270. Springer, Heidelberg (2011)
15. Zagoruiko, N.G., Kutnenko, O.A.: Recognition Methods Based on the AdDel Algorithm. Pattern Recognition and Image Analysis 14, 198–204 (2004)
16. Zagoruiko, N.G., Kutnenko, O.A., Ptitsyn, A.A.: Algorithm GRAD for selection a informative genetic features. In: International Moscow Conference on Computational Molecular Biology, Moscow, Russia, pp. 8–9 (2005)
17. Schlesinger, M.I.: On spontaneous dividing of patterns. In: Reading Automation and Pattern Recognition, pp. 46–61. Naukova Dumka, Kiev. (1965) (in Russian)
18. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: 5th Berkley Symposium on Mathematical Statistic and Probability, vol. 1, pp. 281–297. University of California Press (1967)

Use of the FRiS-Function for Taxonomy, Attribute Selection and Decision Rule Construction

Irina A. Borisova, Vladimir V. Dyubanov,
Olga A. Kutnenko, and Nikolay G. Zagoruiko

Institute of Mathematics of the Siberian Branch of the Russian Academy of Sciences,
pr. Koptyug 4, Novosibirsk, 630090, Russia
biamia@mail.ru, vladimir.dyubanov@mail.com,
{zag,olga}@math.nsc.ru

Abstract. The task of simultaneous taxonomy (task **S**), decision rule construction (task **D**) and most informative attributes selection (task **X**) is the combined-type task **SDX**. We offer a way to solve this type of task with a function of rival similarity (FRiS-function). As a result the set of analyzed objects is divided into K classes (clusters) in the selected subspace of informative attributes according to principles of natural classification. Every cluster is described by a necessary and sufficient set of typical representatives (stolps), which provide maximal similarity of all objects of the training dataset with the nearest stolps. In this paper advantages of the criterion based on the FRiS-function for solving **SDX** task and other combined-type problems in data mining are shown.

Keywords: Pattern recognition, function of rival similarity, clustering, taxonomy, selection of standards, attributes selection, informativeness and suitability of attributes.

1 Introduction

The problems of data mining DM are often formulated starting with an extensive dataset A where M objects are described by N attributes. We are interested in finding out any laws and regularities hidden in this dataset to answer such questions as: What is the structure of these data? Are there compact groups of objects (clusters)? If so, which attributes should be used to distinguish one group from another? How do we classify new objects?

A number of DM methods and algorithms are developed to answer these questions. Groups of objects similar to each other (result of task **S**) are formed by algorithms of clustering or taxonomy. For this purpose it is necessary to know informative attributes (result of task **X**) and a type of a decision rule which will be used for further recognition of the built classes. The informative subset of attributes (result of task **X**) can be found with the greedy search, if classification (result of task **S**) and type of decision rule are chosen. If the system of classes (result of task **S**) and the set of attributes (result of task **X**) are fixed,

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 256–270, 2011.

one can construct a decision rule (task **D**) to classify new objects. There are many different algorithms which would accomplish this.

There are situations when it is necessary to find not only one of elements **S**, **D** or **X**, but two of them: to construct a classification (task **S**) and a decision rule (task **D**) in a given attribute space; to build a decision rule (task **D**) and to select attributes (task **X**) for a given classification; to select attributes (task **X**) and to construct a classification (task **S**) for an established type of a decision rule. In that way, three tasks of combined types **SD**, **DX** and **SX** are formulated.

If the results none of these three tasks (**S**, **D** and **X**) are known, it is necessary to solve a major task of combined type **SDX** – forming the classification (task **S**) in the most informative subspace of attributes (task **X**) with simultaneous construction of decision rule (task **D**) [1].

This paper presents algorithms to solve these combined type tasks. The methods for their decision are based on the use of the function of rival similarity (FRiS-function) which simulates human abilities to estimate similarity between objects or phenomena. This function was successful in solving DM problems in cases where classical statistical methods can't be applied; for example, in tasks where the number of analyzed objects M is less than the number of describing attributes N, where prior information about probability distributions and dependences between attributes is absent.

Let's begin with the description of the FRiS-function.

2 Definition of the Function of Rival Similarity

In many DM problems the concept of "similarity" or "affinity" is used [2]. Every decision rule for recognition of a new object is based on a measure of the "similarity" of the object to a pattern. In taxonomy, objects are united in groups of similar objects. But formal measures of "similarity" cannot be defined correctly without consideration of the context. So, Moscow and Washington will appear "relatives", if one compares the distance between them with Moscow's distance to the sun, but "far" if two cities within one state are considered to be similar. Taking the context into account implies consideration of the competitive situation in some measures of similarity. So, in case of two patterns by the nearest-neighbor rule the decision that object z belongs to some pattern is accepted, when the distance $r_1(z)$ from z to this pattern is not just "small", but when it is less than the distance $r_2(z)$ to the competing pattern. In this case as a distance from an object to a pattern we use the distance from the object to its nearest neighbor from this pattern. Thus, to estimate the similarity of the object z to a pattern, it is necessary to know not only the distance $r_1(z)$ to this pattern, but also the distance $r_2(z)$ to the nearest competitor, and to compare these distances in an ordered scale. To measure the rival similarity in the absolute scale we use the normalized value:

$$F(z) = (r_2(z) - r_1(z))/(r_2(z) + r_1(z)).$$

We call it the Function of Rival Similarity or FRiS-function.

Function F seems to be well coordinated with the mechanisms of the perception of similarity and distinction which humans use, comparing a certain object with two other objects. The value characterizes the similarity of object z with pattern A_1 in competition with pattern A_2. F ranges from -1 to 1. $F = 1$ if object z coincides with an object of A_1, and -1 if object z coincides with an object of pattern-competitor A_2. In the case of identical similarity of the object to both patterns ($r_1(z) = r_2(z)$), function $F = 0$, and that defines the border between patterns.

First we will give a formal definition of the function of rival similarity which is used when we are dealing with labeled datasets (divided into classes).

Let dataset A consists of M objects, for every object values of N described attributes are given. Some metric ρ is setted which allows defining a distance between any two objects of the dataset. Moreover each object is related to one of K patterns, i. e. $A = \bigcup_{i=1,\ldots,K} A_i$, where A_i is a subset of samples of pattern i. Every pattern i is described by the set of k_i typical representatives (stolps) $S_i = \{s_{i_1}, s_{i_2}, \ldots, s_{i_{k_i}}\}$, which is used for the calculation of the distance r from an object z to this pattern: $r(z, A_i) = r(z, S_i) = \min_{j=1,\ldots,k_i} \rho(z, s_{i_j})$ as a distance from z to the nearest stolp of the pattern. The set of all sets of stolps of the K patterns $S = \{S_1, \ldots, S_K\}$, can be treated as a brief description of the dataset A.

With respect to the set S of all K sets of stolps we now introduce for any object a the value $F(a, S)$ of the FRiS-function: let i^* be the pattern of a, i. e. $a \in A_{i^*}$; let $r_1(a, S) := r(a, S_{i^*})$ is a distance to the "own" pattern and $r_2(a, S) = \min_{i=1,\ldots,K, i \neq i^*} r(a, S_i)$ is a distance to a nearest competitor, then the FRiS-function for the object a is calculated by the formula:

$$F(a, S) = (r_2(a, S) - r_1(a, S))/(r_2(a, S) + r_1(a, S)). \tag{1}$$

We use the following average value of the FRiS-function:

$$\overline{F}(S) = \frac{1}{M} \sum_{a \in A} F(a, S) \tag{2}$$

to describe the quality of the set S: The greater $\overline{F}(S)$ is, the better the dataset A is described.

For a new unlabeled object z a nearest pattern i^* is considered as its "own" pattern, so $r_1(z, S) = r(z, S_{i^*}) = \min_{i=1,\ldots,K} r(z, S_i)$.

When we work with an unlabeled dataset A which objects are not divided into patterns (classes), the definition of rival similarity is changed.

Let all objects of set A belong to one pattern. After a set of stolps of this pattern $S = \{s_1, s_2, \ldots, s_k\}$ is fixed, for each object $a \in A$ it is possible to find distance $r_1(a, S)$ (from the object to the nearest stolp from set S). But the absence of a class-competitor does not allow to calculate distance r_2 (from the object to the nearest stolp of a competing class). Because of this, at the first stage a virtual class-competitor is defined. The nearest (virtual) stolp of this class is placed on the fixed distance r_2' from each object of the analyzed dataset. Here r_2' is chosen to be equal to half of the minimal distance between two objects.

Thus, in a classification-construction (taxonomy) task we use some modification of the FRiS-function, which for object $a \in A$ looks like:

$$F'(a,S) = (r'_2 - r_1(a,S))/(r'_2 + r_1(a,S)).$$

Later in the text this modification will be called the reduced FRiS-function (rFRiS). Knowing values $F'(a,S)$ for all M objects of set A, it is possible to calculate the average value of the reduced FRiS-function:

$$\overline{F}'(S) = \frac{1}{M}\sum_{a\in A} F'(a,S). \tag{3}$$

The value $\overline{F}'(S)$ achieves the maximum if stolps from the set S are located in the centers of local accumulations of objects. We used the value of the average reduced function of rival similarity as an indicator to search centers of local accumulations for solving problems connected with the grouping of objects.

After the rival similarity for cases of labeled and unlabeled data has been defined, we will pass directly to the description of how FRiS-functions can be used for the decision of various DM problems.

3 Use of the FRiS-Function for Selecting Stolps of Classes

In our algorithm the process of building decision rules consists in selection of set of typical representatives (set of stolps), which will be used for recognition of new objects [3]. The algorithm of selecting stolps should be independent of the types of the probability distributions of the patterns. The decision it finds out should be adaptable to any situation. In the case of unimodal distributions, the stolps should settle down in the centers of gravity of the patterns. If the distributions are polymodal and the patterns are not linearly divisible, the stolps should stand in the centers of local accumulation of samples. While distributions get more complex the number of stolps should increase.

Algorithm FRiS-Stolp possesses such properties. It is intended to select the minimal number of stolp which protect all training samples from incorrect classification during cross-validation.

If a dataset A with M objects is partitioned into K classes, $A = \bigcup_{i=1,\ldots,K} A_i$, we solve at first the recognition task "the first pattern against all others", where the set of samples of the first pattern is $A_1 = \{a_1, \ldots, a_{K_1}\}$ and the set of samples of the united pattern is $B_1 = \bigcup_{i=2,\ldots,K} A_i = \{b_1, \ldots, b_{M-K_1}\}$.

1. First object a_1 of A_1 is tested as a single stolp of the first pattern. For each objects $a_j \in A_1$ we calculate the distance r_1 from a_j to the stolp a_1 and the distance r_2 from a_j to the nearest object of the rival pattern B_1. Based on these distances the FRiS-function $F(a_j, \{\{a_1\}, B_1\})$ for the object a_j is calculated with the formula (1). The sum of the FRiS-functions over all $a_j \in A_1$: $Q_1(a_1) = \sum_{a_j \in A_1} F(a_j, \{\{a_1\}, B_1\})$ characterizes the protective abilities of the object a_1 as a stolp.

2. For each objects $b_j \in B_1$, the distance r_2 from b_j to the stolp a_1 and the distance r_1 from b_j to the nearest object of the B_1 are calculated. Based on these distances the FRiS-function $F(b_j, \{B_1 \backslash \{b_j\}, \{a_1\}\})$ for the object b_j is calculated with the formula (1). The sum of the FRiS-functions over all $b_j \in B_1$: $Q_2(a_1) = \sum_{b_j \in B_1} F(b_j, \{B_1 \backslash \{b_j\}, \{a_1\}\})$ characterizes the tolerance of the object a_1 to the objects of the rival pattern. Value $Q(a_1) = (Q_1(a_1) + Q_2(a_1))/M$ is used as a measure of efficiency of the object a_1 as a single stolp of the first pattern.

3. Steps 1-2 are repeated for each object a_i of the first patter, $i = 2, \ldots, M_1$, considering as a single stolp of this pattern, one after the other. We find out object a_{i^*} with the maximum value $Q(a_{i^*})$ and it is declared to be the first stolp s_{11} of the first cluster C_{11} of the first pattern. Objects with values of the FRiS-function, exceeding some threshold $F^* \geq 0$ belong to this cluster, i. e. $C_{11} = \{a_j \in A_1 : F(a_j, \{\{a_{i^*}\}, B_1\}) > F^*\}$.

4. All objects of the first cluster are excluded from the list of samples of the first pattern. For the other objects of the first pattern the next stolp is constructed using the steps 1-3. The process lasts until all objects of the first pattern are included in clusters. As a result we obtain k_1 stolps of the first pattern.

5. All objects A_1 of the first pattern are restored.

6. Steps 1-5 are repeated for each task "pattern i against all other patterns", $i = 2, \ldots, K$.

After the completion of all these procedures, the average value of the rival similarity $\widetilde{F_s}$ of all $(M - k)$ objects of the training dataset not selected as stolps is calculated:

$$\widetilde{F_s} = \frac{1}{M-k} \sum_{a \in A, a \notin S} F(a, S), \tag{4}$$

where $k = \sum_{i=1,\ldots,K} k_i$ is the number of stolps of dataset A. This value can be used as a measure of the quality of training.

The process of recognition of control objects is very simple. A new object z is classified as an object of a pattern with a stolp of minimal distances r_1 to z.

We must remember the importance of parameter F^*, mentioned earlier. For small values of threshold F^* (for example, at $F^* = 0$) the number of stolps is small. The minimum number of stolps is equal to the number of patterns K (one standard for the pattern). We are usually satisfied with the simplest variant of the decision rule construction. But care is needed when the average measure of similarity of objects to the stolps is not high.

Increasing F^* leads to an increase in the number of stolps k, which settle down in the centers of local compactness of the distribution. It allows the restoration of the model of distribution more precisely, to find the subclasses which make up the class. Stolps are typical representatives of each subclass. With increasing k the average value of similarity $\widetilde{F_s}$ of objects of the training dataset with standards calculated by (4) grows.

If, however, threshold F^* is equal 1 the number of stolps aspires to match the number of objects M. To warn the algorithm of this extreme measure we

define the penalty function $G = (M - k)/(M - K)$. So, quality of training Q_s we will estimate: $Q_s = \widetilde{F_s} \times G$. Changing parameter F^*, it is possible to find a compromise between the simplicity of a decision rule and the accuracy of the description of the model of distributions.

4 FRiS-Function as Criterion for the Choice of an Informative Subset of Attributes in Problem DX

The combined type task **DX** (recognition with a simultaneous informative attribute selection) has been used in the field of pattern recognition for a long time. Statistically well-grounded methods, however, are developed only for a class of "good" tasks. If they are used in tasks where the number of objects is insignificant and less than the number of describing attributes, the unequivocal answer to the question of how effective selected subsystems will appear at recognition of control dataset does not exist. With an increasing number of describing attributes (for a fixed number of objects of the dataset), the probability of occurrence of pseudo-dependencies between some of these attributes and the target attribute increases. As a result, irrelevant attributes can be selected as informative ones [4]. This can lead to serious mistakes in the recognition of a control sample. Our researches have shown that criterion $\overline{F}$, based on the FRiS-function is more effective in such situations than the following existing analogues.

In the majority of existing methods, the rate U of objects of training dataset correctly recognized by the KNN rule, estimated by the method One-Leave-Out, is used as a feature subsystem quality.

Another criterion which presumably allows a good choice of informative subsystems of attributes is based on Fisher's idea to estimate informativeness through the ratio where the distance between sample means of patterns is divided by the sum of their variances:

$$Q = |\mu_1 - \mu_2|/(\sigma_1 + \sigma_2).$$

The coordinates of the center of gravity of the pattern objects are used as the estimated value of a sample mean of a pattern. An average square of the distances from the objects of a pattern to its center of gravity is used as a variance.

The next considered measure of informativeness is interesting for us because of the use of distances from each object a of the training dataset to the nearest "one's own" pattern $r_1(a)$ and the nearest "competitor" pattern $r_2(a)$ in it. So it could act like the function of rival similarity and give results that are not bad. This method of calculation is described in the family of algorithms entitled RELIEF [5]. There are various forms of this method. The direct criterion of attributes informativeness W, used in our researches, looks as follows:

$$W = \frac{1}{M} \sum_{a \in A} \frac{r_2(a) - r_1(a)}{r_{max} - r_{min}}.$$

With this criterion, as well as with FRiS-functions, the difference between distances to the nearest competitor and to the nearest "one's own" patterns is

calculated, and for normalization the difference between maximal distance r_{max} of objects of the dataset in the analyzed feature subspace and the corresponding minimal distance r_{min} is used.

The simplest informativeness criterion F_s, based on the FRiS-function is calculated by formula (2) on the assumption that all objects of the training dataset, besides the one for which the value of FRiS-function is calculated, are the stolps of the corresponding patterns.

These four criteria – ratio of correctly-recognized objects of the training dataset (U), average value of function of rival similarity (F_s), Fisher's criterion (Q) and RELIEF criterion (W) – were compared in the following modeling experiment.

Initial data consisted of 200 objects of two patterns (100 objects of each pattern) in a 100-dimensional space. Attributes were generated so that they possessed different informativeness. As a result, about 30 attributes appeared to be to some extent informative, and other attributes were generated by the random-number generator and were obviously not informative. For training, 35 objects of each pattern casually got out. On the control, the 130 other objects were recognized. At each task of the algorithm AdDel [6] the best n-dimensional subsystems of attributes (n from 1 to 22) got out. The share of correctly-recognized control objects for every subsystem was calculated. Results are presented in Fig. 1.

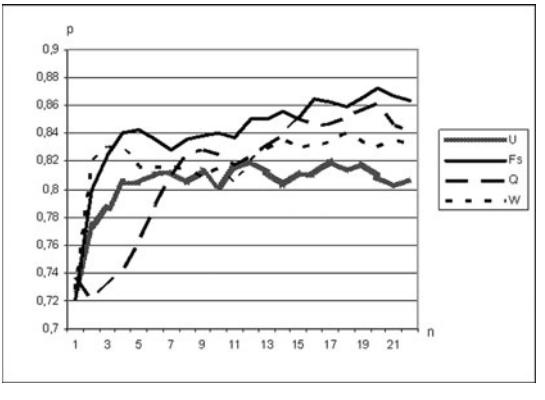


Fig. 1. Reliability of recognition of control dataset in various subsystems of the informative attributes selected by one of four criteria: share of misrecognized objects (U), function of rival similarity (F_s), Fisher's criterion (Q) and criterion from algorithm RELIEF (W)

5 FRiS-Function in a Taxonomy Task

In this task we deal with a situation where there are no class labels for the training dataset A. Clearly, in this case we should use the reduced function of rival similarity, as defined in section 2. We are reminded that in this modification the virtual pattern-competitor is settled on the fixed distance equal r'_2 to all objects of the dataset.

On the basis of FRiS-functions we developed an algorithm of taxonomy FRiS-Tax [7] to simulate the work of a human expert constructing classification. This algorithm consists of two stages. At the first stage, named FRiS-Cluster, the centers of local accumulation of objects are found. Each of them is considered as the stolp of the corresponding cluster. All objects of the dataset are naturally distributed between clusters. An object a is grouped into that cluster whose stolp is the nearest one. The obtained grouping can be considered as a final result, if the expert accepts its quality. Otherwise the results of clustering pass to the second stage of the algorithm, named FRiS-Class. At this stage runs the procedure of forming taxons, which have a more complex structure. Clusters, satisfying certain conditions, are considered to belong to one class. It allows the creation of taxons which have free forms, and cannot be linearly distinguished.

5.1 Clustering (Stage FRiS-Cluster)

The described algorithm defines the number of clusters automatically. A user sets only the maximal number of clusters k_{max}. The algorithm searches for decisions of a task for all numbers of clusters $k = 1, 2, \ldots, k_{max}$ consistently, to choose from them the most successful decision. If a dataset A of M objects without labels is given as input, it acts as follows:

1. $k = 1$. A randomly-chosen object a is appointed as a stolp, and under this condition the average reduced FRiS-function $\overline{F}'(\{a\})$ over the dataset A is calculated with formula (3).
2. Step 1 is repeated for all M objects of A. Every object is consequently appointed as a stolp. As the first stolp s_1 the object that has maximal value $\overline{F}'(\{s_1\})$ is picked out. This stolp is considered as typical representative of cluster $C_{11} = A$. The clustering quality for $k = 1$ is $F_c(1) = \overline{F}'(\{s_1\})$.
3. $k = 2$. After the first stolp s_1, we take a randomly-chosen object a, different from s_1, as a candidat for a second stolp. Then for the set of two stolps $\{a, s_1\}$ the average reduced FRiS-function $\overline{F}'(\{a, s_1\})$ is calculated with formula (3).
4. Step 3 is repeated for all objects of the dataset A which are different from s_1. As the second stolp s_2 the object, which in pair with s_1 provides the maximal value of the average reduced function of rival similarity $\overline{F}'(\{s_2, s_1\})$, is picked out.
5. After the detection of two stolps, the dataset A is partitioned into clusters C_{21} and C_{22}, by the following rule. An object a is grouped into that cluster with a stolp of minimal distance r_1 to a . So $C_{21} = \{a \in A : \rho(a, s_1) \leq \rho(a, s_2)\}$, $C_{22} = A \backslash C_{21}$. Stolp s_1 provides maximal average similarity for the objects of the cluster C_{11}, but for the descrypting of the cluster C_{21} in competition with the cluster C_{22} some other object from the cluster C_{21} can be best according to the average FRiS-function. To find the final position for the stolps we carry out the following procedure.
6. A randomly chosen object $a_1 \in C_{21}$ is appointed to be a stolp of this cluster, and the average FRiS-function $\overline{F}(\{a_1, s_2\})$ is calculated with the formula (2).

7. Step 6 is repeated for all objects of C_{21}. The object s_{21} which provides the maximal value of the average FRiS-function $\overline{F}(\{s_{21}, s_2\})$ is picked out as a new stolp of C_{21}.
8. A new stolp of the cluster C_{22} is determined by repeating steps 6-7 for all objects of the cluster C_{22} and an object s_{22}, that provided maximal value of the function $\overline{F}(\{s_{21}, s_{22}\})$ is selected. In Steps 6-8 we use the FRiS-function that imitates the process of a competition between real stolps instead of the reduced FRiS-function. The clustering quality for $k = 2$ clusters is $F_c(2) = \overline{F}(\{s_{21}, s_{22}\})$.
9. For the further increasing the list of stolps we use the stolps s_1 and s_2 selected with the reduced FRiS-function. As in Steps 3-4 all objects of the dataset are checked one by one for the role of the third stolp. For each variant the average reduced function of rival similarity is calculated and the object s_3, for which this value $\overline{F}(\{s_1, s_2, s_3\})$ is maximal, is fixed as the next stolp. All objects of A are redistributed among the three clusters C_{31}, C_{32} and C_{33}. $C_{3i} = \{a \in A : \rho(a, s_i) \leq \rho(a, s_j), j = 1, \ldots, 3, j \neq i\}$, $i = 1, \ldots, 3$. Then redefinition of stolps positions is done taking into account rival situation, as described in Steps 6-8, and the quality of clustering $F_c(3)$ is calculated.
10. The process proceeds until clustering for all numbers of clusters $k = 1, 2, \ldots, k_{max}$ has been obtained. During the process the preliminary list of stolps $\{s_1, s_2, \ldots, s_{k-1}\}$ is used for finding the next stolp s_k, and for determining final grouping $\{C_{k1}, C_{k2}, \ldots, C_{kk}\}$ and estimation of its quality $F_c(k)$. We emphasize that passing from reduced to usual FRiS-function, as well as reinstalling stolps, should be realized only for the detection of final positions of k stolps.

It is obvious that if all objects of dataset A are used as stolps (dataset is described by M stolps), the clustering quality $F_c(M)$ reaches the maximal value equal to 1. On the way to this global extremum, however, there are local ones. Our experiments have shown that local maxima occurs for such numbers of clusters that human experts regard regards as "reasonable" in the sense that objects which are grouped into different clusters by the expert, are also grouped into different clusters by our algorithm.

5.2 Construction of Classification (Stage FRiS-Class)

Often classes formed by experts have a complex structure that can't be described accurately by a single objects (stolp) for a single class. Therefore we have introduced the stage FRiS-Class in the algorithm FRiS-Tax. On this stage a procedure for the association of several clusters in one class is realized. It is started for the most successful variants of clustering, got out on the first stage.

The basic idea underlying this stage of the algorithm consists of the following. If the clustering stage is successful, clusters composed of different classes are separated from each other by zones with a lowered density of objects, and near the borders of clusters comprising one class such downturns of density are not presented. The objects of the dataset are distributed there in a regular way. If we have clustering $\{C_1, \ldots, C_k\}$ the formalized algorithm for checking object distributions near borders of clusters looks as follows:

1. For any two different clusters C_i and C_j the objects which are located in a zone of competition between these clusters are close to the border among the two clusters. An object $a \in C_i$ is considered to be in the zone of competition for clusters C_i and C_j, if the following conditions are satisfied:

- Two nearest to the object a stolps are the stolps s_i and s_j of clusters C_i and C_j;
- The value of similarity $F(a, \{\{s_i\}, \{s_j\}\}$ of this object with cluster C_i in competition with C_j is less than some threshold $F^* > 0$;
- The distance from a to the stolp s_i is less than the distance between stolps s_i and s_j. This condition is necessary in cases when a considerable distance to the stolp can mean the remoteness of the object from all stolps.

Those pairs of clusters whose zones of competition are not empty are considered as candidates for uniting.
2. In clusters C_i and C_j we select two objects $a \in C_i$ and $b \in C_j$ from the zone of competition of these clusters, such that the distance between a and b is minimal.
3. For the object a its nearest neighbour a' from the cluster C_i and the object b its nearest neighbour b' from the cluster C_j are found.
4. Clusters C_i and C_j are united into a single class, if the values of $\rho(a, b)$, $\rho(a, a')$ and $\rho(b, b')$ differ from each other only slightly. For example, the following condition can be checked:

$$\rho(a, a') < \alpha\rho(b, b') \& \rho(b, b') < \alpha\rho(a, a') \& \rho(a, b) < \alpha(\rho(a, a') + \rho(b, b'))/2, \alpha > 1$$

After termination of the second stage the quality of classification is recalculated. But now, contrary to the first stage when every cluster was described by a single stolp, more complex structures – the classes described by a set of stolps are considered.

5.3 Choice of an Optimum Number of Taxons

Let's remember that before starting the algorithm the user sets a range in which the number of clusters k varies. After the termination of the first stage we have one variant of clustering for every number of clusters k from the range with the calculated clustering quality $F_c(k)$. At the second stage of the algorithm we already work only with those variants of clustering which appeared locally-maximal, i. e. $F_c(k-1) < F_c(k)) \& (F_c(k+1) < F_c(k)$.

After the second stage coincident variants of classification, and variants which, as a result of uniting, form a single class, are eliminated. But even after that there can be some variants of taxonomy from which we have to choose the best one. Our experiments have shown that we have a few variants of taxonomy which have an insignificant difference between numbers of clusters, and the best one is that taxonomy which provides the maximal average value of the FRiS-function. In case of a big difference between numbers of clusters in different variants this approach cannot be used. Such "ambiguity", however, will correspond with the following observation. Depending on the level of detail demanded, several

variants of taxonomy with different numbers of classes can be constructed by a human, too. For example, in a military structure taxon-regiments and taxon-battalions can be allocated, and to tell whether one of these variants is better than another is impossible until the purpose for which the taxonomy is created is strictly fixed.

In Fig. 2 four test examples are shown for which it was possible to determine the optimum number of clusters and classes by the algorithm FRiS-Tax.

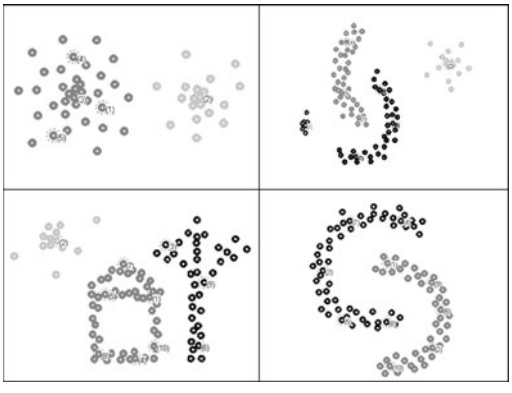


Fig. 2. Examples of the tasks successfully solved by the algorithm FRiS-Tax

6 FRiS-Function in Problem SX as a Criterion for Natural Classifications

The algorithm described above solves a problem of clustering construction in a fixed space of attributes. If, for the description of the same objects another feature subset is used, the results of clustering can change. So it is clear that the process **S** of clustering is directly connected with the process **X** of attribute selection. There are some questions to be answered: When should the problem of combined type **SX** be solved instead of problem **S**? Which properties of classification should be provided by the selection of the attribute subsystem?

We can search for the answers to these questions by analyzing specific properties of so-called "natural" classifications [8]. Such classifications were created by outstanding scientists of the past and have been used successfully until now. As examples of natural classifications, Mendeleev's classification table, Linne's biological classification, Darwin's species classification and Fedorov's classification of crystals stand out. These classifications have a number of interesting properties: [9]

- as a rule, natural classifications have hierarchical structures – they have classes, subclasses and so on;
- division of classes into subclasses is based on a small number of attributes;

– at each level of the hierarchy of classification every cluster in the subspace of these attributes should satisfy the "geometrical" requirements usually formulated in many algorithms of clustering analysis – high similarity of objects into one class and big differences between objects from different classes.

There is one more important feature: natural classifications possess high inductive ability. It means that a few ("essential") attributes, on which classification is made, allows the prediction of some other ("latent") properties of objects of the given class. A classical example of inductive ability of classification is classification of chemical elements by their nuclear weight in Mendeleev's table, where one attribute predicts many physical and chemical properties of those elements.

Thus, a classification possessing the property of "naturalness", on the one hand should satisfy geometrical criteria, controlling quality of grouping, and, on the other hand, should possess inductive ability and predictive force. We assert that the inductive ability of natural classifications is the most important property. Therefore, natural classifications is a very desirable result for the process of structuring information. The average value of the FRiS-function, calculated by formula (2), can be used as a criterion which simultaneously allows us to trace geometrical "compactness" of taxons in space of essential attributes, and to estimate predictive ability of taxonomy for the attributes which are not essential.

7 FRiS-Function in the Problem SDX

After several algorithms based on FRiS-functions have been constructed for simpler combined-type problems, we can pass directly to the problem **SDX**, the most complex problem of data mining – the problem of information structuring and ordering. One interpretation of this task is data compression into a form suitable for further analysis and usage by humans. It means the compressed dataset should consist of a small number of groups of objects (**S**), described in a small informative subspace of initial attributes (**X**). This description should contain a system of simple decision rules (**D**), according to which each new analyzed object can be classified. The reduced description of the dataset should also contain a system of simple rules to predict values of the attributes, which have not entered into an informative subset. In other words, these rules allow the restoration of values of all attributes by values of informative attributes; they also allow the restoration of the general characteristics of the class of a given object.

Let's consider a variant of this problem when each group of objects is described by typical representatives (stolps), and as predicted values of attributes of a new object their values for the stolp nearest to this object in the space of informative characteristics are used. For an estimation of reliability of such forecasts we use the function of rival similarity.

Let A be the initial set of objects, and Y be the initial set of attributes. For some set of stolps $S \subseteq A$ and some set of informative attributes $X \subseteq Y$ we define

the quality $Q_F(S, X)$ of the description of the initial set $\langle A, Y\rangle$ by the selected dataset $\langle S, X\rangle$ in the following way:

$$Q_F(S, X) = \sum_{a\in A} F_Y(a, s_a^*), \text{ where } s_a^* = \arg\min_{s\in S} \rho_X(a, s).$$

Here F_Y is a function of rival similarity in space Y, ρ_X is the distance in the subspace X, s_a^* is a stolp of S which is nearest to the object a.

The problem consists in finding such a pair $\langle S, X\rangle$ which will provide a maximum value of Q_F. To obtain approximate solutions to this challenge we divide it into two simpler tasks and consider a problem of two-level optimization:

$$Q_F(S_X, X) = \sum_{a\in A} F_Y(a, s_a^*) \to \max_{X\subseteq Y}, \text{ where } s_a^* = \arg\min_{s\in S_X} \rho_X(a, s),$$

$$S_X = \arg\max_{S\subseteq A, |S|\le m^*} \sum_{a\in A} F_X(a, s_a^*), \text{ where } s_a^* = \arg\min_{s\in S} \rho_X(a, s).$$

Here m^* is the maximum number of stolps in S. The set of stolps S_X in the fixed subspace of attributes X is found by the algorithm of taxonomy FRiS-Tax, which builds a set of stolps providing the maximal of the average value of the function of rival similarity on the dataset. For finding an informative subsystem $X^* = \arg\max_{X\subseteq Y} Q_F(S_X, X)$, one of the existing procedures of feature selection can be used.

Elaboration of this idea allows us to get solutions to the task **SDX** which has a hierarchical structure presented in the form of a tree T [15]. Each division of the parent nodes of this tree corresponds to some subset of objects clustering in a special informative subspace of features. Each leaf node of this tree is a low-level class in the created classification. The stolps for all low-level classes are defined in the united space of attributes X used at all levels of the classification on the tree T. During the recognition of some new object a by the tree T the corresponding node (class) is found step-by-step by moving from the root node to the leaf nodes, and the function of rival similarity of this object with a stolp of this class $(T(a))$ in the full space of attributes Y is calculated. The optimized criterion for the quality of the construction of a tree in this case is:

$$\tilde{Q}_F(T) = \sum_{a\in A} F_Y(a, T(a)).$$

The approximate solution for this problem is found through transition to a series of tasks. Each simple task in this case corresponds to dividing one leaf node in a tree that provides maximal increase of the global quality $\tilde{Q}_F$.

Algorithm FRiS-SDX has been applied to the analysis of the following real task in order to construct a solution to the hierarchical **SDX** problem. The spectral characteristics of samples of various minerals have been measured. Each spectrum was represented by a 1024-dimensional vector. Spectra of 160 samples

from a training dataset were divided by experts into seven groups on the basis of their chemical structure. Efficiency of the algorithm was estimated through the analysis of the value of uniformity which taxons received from the perspective of the chemical structure of the objects which were in them. We didn't use our knowledge of the chemical structure of samples during the work of the algorithm FRiS-SDX, except for an estimation of the success of the results.

The algorithm divided each parent node of this tree into two classes, described by two stolps. Informative subset for clustering should not contain more than six attributes (i. e., spectral strips) from 1024. The tree received as a result of the work of the algorithm with these restrictions, is represented in Fig. 3. It has consistently separated four classes of chemical substances from each other, and put three in one class. More detailed analysis has shown that the samples which have been put into one class have practically identical chemical structure. The traces of manganese and cobalt are so negligible that their spectral portraits are indistinguishable.

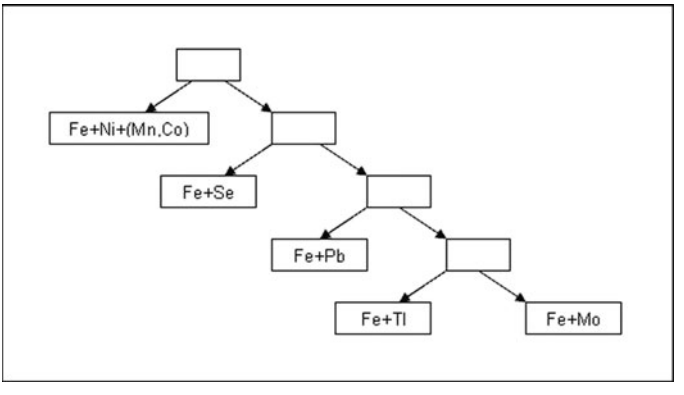


Fig. 3. The structural tree constructed by the algorithm FRiS-SDX

8 Conclusion

The universal approach to the solution of data mining problems of basic and combined types, based on the use of the function of rival similarity (FRiS-functions) is described. On this basis it is possible to build effective algorithms, invariant to the ratio of the number of objects to the number of attributes in the dataset and to the type of the probability distribution of the samples. The solution to the main task of the combined-type **SDX** of the simultaneous construction of classification (task **S**) of observable objects, building decision rule (task **D**) and selection of informative subset of attributes (task **X**) is shown.

Acknowledgments. This work has been done with the support of the Russian Federal Property Fund (grants 05-01-00241 and 08-01-00040).

References

1. Zagoruiko, N.G.: Pattern Recognition methods and their using. Soviet Radio, Moskow (1972) (in Russian)
2. Voronin, J.A.: Beginning of Theory of Similarity. Computer Centre SD RAS, Novosibirsk (1989) (in Russian)
3. Zagoruiko, N.G., Borisova, I.A., Dyubanov, V.V., Kutnenko, O.A.: Methods of Recognition Based on the Function of Rival Similarity. Pattern Recognition and Image Analysis 18, 1–6 (2008)
4. Borisova, I.A., Zagoruiko, N.G., Kutnenko, O.A.: The Criterion of Informativeness and Suitability Subset of Attributes Based on Function of Similarity. Zavodskaja Laboratorija 74, 68–75 (2008) (in Russian)
5. Kira, K., Rendell, L.: The Feature Selection Problem: Traditional Methods and a New Algorithm. In: 10th National Conference Artificial Intelligence (AAAI 1992), pp. 129–134 (1992)
6. Zagoruiko, N.G.: Applied Methods of Data and Knowledge Analysis. In: Institute of Mathematics SD RAS, Novosibirsk (1999) (in Russian)
7. Borisova, I.A.: Clustering algorithm FRiS-Tax. Scientific bulletin of NGTU 3, 3–12 (2007) (in Russian)
8. Vityaev, E.E.: Algorithm of Natural Classification. Computer Systems 99, 44–50 (1983) (in Russian)
9. Zagoruiko, N.G., Borisova, I.A.: Principles of natural classification. Pattern Recognition and Image Analysis 15, 27–29 (2005)

Similarity Determination for Clustering Textual Documents

Vladimir Barakhnin, Vera Nekhaeva, and Anatolii Fedotov

Institute of Computational Technologies SB RAS, Novosibirsk State University
Lavrentyeva 6, 630090 Novosibirsk, Russia
bar@ict.nsc.ru, nekhaeva@ngs.ru, fedotov@sbras.ru
http://www.ict.nsc.ru

Abstract. The problem of computerized selection of textual documents on scientific subjects is treated with new improved methods; that could be of interest for an individual researcher or a research team. Attributes of a bibliographical description (authors, keywords, abstract) are proposed to be used as scales for the measure determination. The values of weight coefficients in the formula for calculating the similarity measure are determined by the assumed a posteriorireliability of the respective scale data.

Three classical document clusterization methods have been analysed in order to find the ones potentially feasible for the solution of the formulated problem: clusterization by finding cliques in the full matrix of documents similarity, clusterization by Rocchio method and the method based on the so-called greedy algorithm as well as the new method suggested by N.Zagoruiko based on employing the function of rival similarity (the so-called FRiS-function). Testing showed that the FRiS algorithm proved to be the most efficient one for this problem although the greedy algorithm also yields acceptable results.

Keywords: similarity, clusterization of textual documents.

1 Introduction

Millions scientific papers are published in the world annually. It is impossible even to browse superficially the publications in narrow fields of science. This accounts for the universal popularity of electronic editions providing information about new publications in particular:

- databases of Abstracting Journals;
- "Current Contents" databases;
- specialized online databases like Zentralblatt MATH.

In addition, each researcher over the years of his professional activity accumulates a card index containing the description of papers, books, etc. of interest to him. In this case, the main selection criterion is the personal interests of the researcher. Nowadays, such card indexes are usually in an electronic form. This

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 271–279, 2011.

makes it possible to create integrated card indexes by pooling the resources of scientists working in collaboration.

Therefore, the problem arises of computerizing the process of selecting publications from electronic databases of interest to a particular researcher or a group of scientists working in collaboration. In order to find the paper he needs, the researcher browses either abstracting journals or their electronic analogues. Since there are sufficiently efficient algorithms for the search of a particular electronic publication, the topical issue today for information retrieval is finding a suitable class of documents with similar content on the basis of a given document.

This work contributes to the solution of the problem of computerizing the process of selecting publications from bibliographic databases, which may be of interest for a specific researcher or a group of researchers working in collaboration. To this end, at the first stage of our work, algorithms for measuring the similarity between two documents of an electronic database, and algorithms of clustering documents in this base are studied. It is suggested that attributes of the bibliographic description in the entry of documents should be used as scales for measuring the similarity.

Under clustering we, according to [1], understand the grouping of a set of documents from an electronic database into classes, such that elements falling under the same class are characterized by a higher similarity than elements falling under different classes (in our case, the description of documents under clustering is based on the attributes of the relevant bibliographic entry). Each cluster thereat is usually described by means of one or several identifiers called a profile or a centroid. The cluster profile can be represented as some formal object located in the center of the cluster or any representative object capable of characterizing the other objects of this cluster (further we will focus on the methods of defining centroids in more detail). Making use of the centroid concept one may look for similar documents by comparing retrieval requests first with the profiles of clusters and then checking the entries inside the clusters with very similar profiles.

2 Similarity Determination for a Set of Documents

As has already been mentioned, as scales for measuring the similarity between two documents, attributes of the bibliographical description of these documents will be used. Let us make a list of key elements of a bibliographic description of documents entered in a card index:

- authors;
- title;
- journal or publishers' name;
- year of issue;
- volume, issue, pages (for publications in periodicals);
- abstract.
- classifier codes;
- keywords.

Below, an algorithm for measuring the similarity of a new document with the documents from the available set (i.e. personal bibliographic base) is presented. The quantitative characteristics of the similarity is determined for the set of documents D as follows:

$$m : D \times D \to [0, 1], \tag{1}$$

therewith the function m in the case of complete similarity assumes the value 1, and, on the contrary, it is equal to 0. The similarity is calculated by the formula of the following form

$$m(d_1, d_2) = \sum \alpha_i m_i(d_1, d_2), \tag{2}$$

where i is the running number of an item (attribute) of the given bibliographic description, α_i designates weighting coefficients whereas $\sum \alpha_i = 1$ (see, for instance, [2]), $m_i(d_1, d_2)$ is the similarity with respect to i-th item (in other words, with respect to the i-th scale). Since in the described situation practically all the scales are nominal, the similarity with respect to the i-th scale is obtained in the following way: if the values of i-th attributes of the documents coincide, the degree of proximity $m_i(d_1, d_2)$ is 1, otherwise it is 0. Therewith it must be taken into account that the attribute values can be aggregated. Then $m_i = n_{i1}/n_{i0}$, where $n_{i0} = \max\{n_{i0}(d_1), n_{i0}(d_2)\}$, and $n_{i0}(d_j)$ is the total number of items constituting the value of the i-th attribute of the document d_j, with n_{i1} denoting the number of coinciding items.

Note that the above-presented algorithm for measuring the similarity may lay the foundation of some expert system with certain condition-action rules. Thus, the value of the weighting coefficients α_i in formula (2) can be determined by the assumed *a posteriori* validity of data of the respective scale. For example, complete (or even "almost complete") coincidence of the value of the attribute "authors" of the document d_1 and document d_2 has higher weight in the case when the number of values of this attribute in the document d_1 is sufficiently great (as compared to the case when the document d_1 has only one author). In this situation we can increase the value of the respective weighting coefficient in formula (2) with simultaneous proportional reduction in the other coefficients.

3 Methods for Documents Clustering

The main problem of documents clustering consists in finding a grouping of the documents, such that the elements of every group are so similar to each other that in some cases their individual features could be neglected. In particular search in a systematized file is much easier than in a non-systematized one because groups of documents whose profiles bear no similarity with the retrieval request are not included in the in-depth retrieval process. Under document clustering it is important to attain a reasonable compromise between the cluster size avoiding both the generation of numerous very small clusters (which impairs the effectiveness of clustering as identification of sets of similar documents) and a small number of very large classes (which will reduce the search precision).

It is conventional to distinguish between some tasks of classification: cluster generation based on the data (properties and characteristics) about the classified objects; objects assigning to the generated clusters or clusters in the process of generation; retrieval of information required for the identification and description of document classes. The generations of classes in itself is usually based on the comparison of document vectors, at the same time a class is defined as a set of all the objects with sufficiently high values of the similarity coefficient. Defining the characteristics of a class is equivalent to the generation of a profile; the assignment of objects to class depends on the degree of similarity between the classifiers of the objects and the profiles of classes.

In our present work we study clustering methods, for which the comparison criteria used are only pre-specified items of a bibliographic description not allowing for individual retrieval options for those documents or consumers' opinion about their feasibility.

As potentially useful for the solution of the problem formulated three classical clustering methods have been analyzed: clustering by finding complete subgraphs in the full matrix of documents' similarity [1], clustering by Rocchio method [1], and the method based on the so-called greedy algorithm [3] as well as a new algorithm based on the use of FRiS functions [4]. Let us dwell briefly on the essence of the above-listed algorithms.

The process of finding maximal complete subgraphs is based on the construction of the full matrix of similarity, by means of which each pair of documents (d_1, d_2) is matched with a similarity coefficient $S(d_1, d_2)$. Usually a threshold value is selected and the similarity matrix is reduced to the binary form by substituting all the similarity coefficients such that $S(d_1, d_2) \geq$ by a unity and all the rest — by zero. Then the sought classes are obtained as the maximal complete subgraphs of the graph described by the binary form of the similarity matrix.

In the Rocchio algorithm, the construction of the similarity matrix is replaced by the check of the space density for some documents (the space density for the document d_i is defined as number of documents dj such that $S(d_i, d_j) \geq$, where a threshold value is selected). Only those documents are treated as the possible centers of the clusters that by calculation results proved to be located in dense space areas. The clustered document is assigned to the class, the similarity with the centroid of which proved to be the highest.

On using the greedy algorithm a row (or a column since the matrix is symmetric) is found in the similarity matrix where the sum of the components will be maximum. The document corresponding to this row is declared as the center of the first cluster and incorporate in it all the documents whose similarity coefficients exceed or equal some pre-specified threshold value. Then all the documents incorporated in the cluster are removed by eliminating the respective rows and columns from the matrix, reiterating the process several times until all the documents have been clustered.

In the clustering method based on FRiS functions on determining the similarity between two documents a rival situation is considered: decision about

document d assignment to the first cluster is made not when the distance r_1 to this cluster is "small" but when it is smaller than the distance r_2 to the rival cluster. In order to calculate the degree of the rival similarity measured on the absolute scale, a standardized value $F_{12} = (r_2 - r_1)/(r_2 + r_1)$ is used, called a FRiS function (Function of Rival Similarity). Obviously, at the first stage of clustering when there are no rival clusters as yet, one has to deal with a certain modification (reduction) of the FRiS-function using a virtual rival cluster. Here the essence of applying this algorithm is that on using a reduced FRiS-function, centers of local "clots" ["bunches"] of document distribution are chosen as the centroids which is followed by the generation of linearly separable clusters.

4 Choice of the Optimal Algorithm

The practical purpose of applying the analyzed algorithms was to computerize the process of selecting publications from electronic databases that could be of interest for a particular researcher or a group of researchers working in collaboration.

The algorithms were tested on the electronic database of the "Siberian mathematical journal" containing bibliographic descriptions of the journal articles published from 2000 to 2005. To the articles in the said database in addition to the standard attributes (title, author, publication year etc.) the respective codes are assigned from the Mathematics Subject Classification (MSC-2000). This fact enabled us to subdivide the entire work into two stages:

1. Finding the optimal clustering algorithm. The earlier defined construction is used as a measure in the document space, the comparison, however, is performed on the basis of a single attribute, i.e. — classifier codes. Since the coincidence of those codes for a group of documents is an objective criterion of the coinciding subjects of those documents, this measure may be considered ideal.
2. Specifying the measure (i.e. obtain the weighting coefficient for each attribute in formula (2)) that after the base clustering yields the result approaching that using the measure determined in Item 1.

The comparison of the three classical algorithms showed that the determination of clusters as maximal complete subgraphs obtained from the similarity matrix proved to be practically inapplicable to the solution of the formulated problem since it tends to generate numerous very small groups. The results obtained by the Rocchio algorithm proved somewhat better: since in this method clustering of selected documents is performed, rather large classes can be obtained. However, the calculation of the space density left the major part of the documents out of all the clusters.

A much more satisfying result was obtained by means of the greedy algorithm. Its use yielded a cluster array where each cluster contains on the average 6-10 entries (for comparison: the total number of articles in the database is about 700). Thereat despite the necessity of constructing a similarity matrix, time consumption was about the same as for those required by the Rocchio algorithm

(the reasons for that are described above in a greater detail). Therefore, the greedy algorithm has several advantages over the method of complete subgraphs and the Rocchio algorithm:

1. We do not have to face the problem of too few too large clusters.
2. We do not have to face the problem of too many small clusters.
3. No documents can be left out of all the clusters.
4. We do not have to face the problem of obtaining the document profiles, i.e. centers around which clusters are formed.

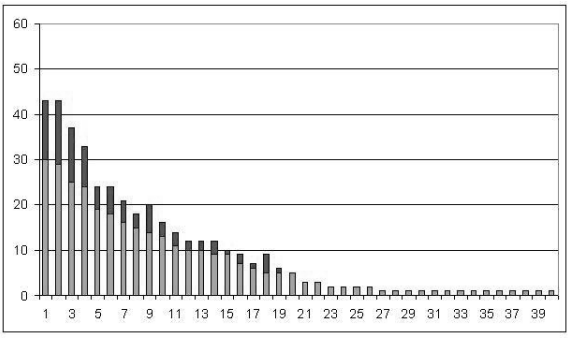


Fig. 1. Greedy algorithm

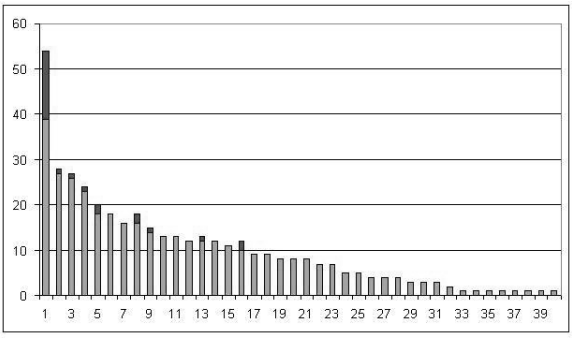


Fig. 2. FriS-algorithm

Then FRiS-algorithm and greedy algorithm were compared. It turned out that FRiS-algorithm gives better clustering accuracy. Below, the results of clustering the databases of "Siberian mathematical journal" are presented obtained by means of greedy algorithm and FRiS-algorithm.

Histograms (Fig. 1 and Fig. 2) show the composition of the obtained clusters. Along the horizontal line, the conventional numbers of clusters are mapped (corresponding to particular sections of the classifier MSC-2000), the vertical axis shows the number of documents in a cluster. As a criterion for the validity test

of inclusion a publication in the cluster, its MSC-2000 classifier code was used. If the classifier codes of the cluster centroid were included in the classifier codes of this entry, we assumed that the entry was correctly included in the cluster.

As can easily be seen, the size of "noise" (shown in the upper part of the columns) in the clusters under clustering by FRiS-algorithm is considerably lower than for the case of the greedy algorithm. Moreover, clustering is more uniform, and the percentage of one-element clusters is significantly lower.

Among comparative disadvantages of the FRiS-algorithm we should name the necessity of manual specification of clusters number in clustering and a little higher computational complexity — $O(kN^2)$ (where k is the user-specified number of clusters) as compared with $O(N^2)$ of the greedy algorithm. However, on clustering large bases such increase in complexity will no longer be so important, furthermore to create a system computerizing the process of scientific publications selection, databases should be clustered only once. Therefore, FRiS-algorithm was recognized as the optimal algorithm for solving the problem of clustering databases of scientific publications.

5 Looking for the Optimum Method for Specifying the Similarity in the Set of Documents

In order to specify the measure in the set of document we used formula (2) where the following attributes of the bibliographic descriptions were used as scales:

- authors;
- key words;
- abstract.

Since the comparison of abstracts in an explicit form (i.e. as text lines) is obviously useless, the problem of singling terms out of the general texts of abstracts was solved as a separate subproblem. At present, a special web-application is accessible generating at a query an xml-document with a list of mathematical terms included in this query (as a source of terms, thesaurus [5] is used, based on the "Mathematic encyclopedia"). Since in the Russian language nouns and adjectives change their form when they are declined, the calculation of the occurrence in the text of phrasal terms from a pre-specified set is a nontrivial problem. The algorithm of the WEB-application is based on the use of two indexes containing triads: "text number" — "position in the text" — "word number in the lexical dictionary" and "term number" — "word position in the term" — "word number in the lexical dictionary". Thereat while the first index occurs in practically every information retrieval system, the introduction of the second index that allows the user to enhance drastically the algorithm efficiency is quite original. The term index is located in the data storage [repository] of the WEB-application together with their list and is supplemented as this list is extended (changed).

Therefore, after the integration of this application into the document-clustring program the user will be able to compare abstracts like other compound attributes. In addition, on specifying the measure, it was taken into consideration that the values of the weighting coefficients in formula (2) are determined by the assumed *a posteriori* validity of data of the respective scale and in some cases one of the coefficients can be increased with the proportional decrease in the others.

In order to obtain the weighting coefficient for each attribute, clustering of samples was performed from databases of the "Siberian mathematical journal". We considered samples of varying capacity, and as the truth criterion the result of clustering was used which had been obtained using the measure based on MSC-2000 codes.

As shown by the experiment, the greatest similarity with the clustering result based on classifiers codes was attained by introducing the following condition-action rules:

1. If each of the documents d1 and d2 has more than two authors and at least 2/3 of their authors coincide, the respective weighting coefficient of the attribute "authors" was assumed to be equal to unity.
2. If each of the documents d1 and d2 contains more than three key words and at least 3/4 of those words coincide, the corresponding weighting coefficient of the attribute "key words" was assumed to be equal to unity.
3. If each of the documents d1 and d2 contains more than four terms from the abstract thesaurus and at least 3/5 of these terms coincide, the weighting coefficient of the attribute "abstract" was assumed to be equal to unity.

Otherwise we assume the coefficient of the attribute "authors" equal 0.2, and of the attributes "key words" and "abstract" it is assumed to be 0.4.

Interestingly, these rules proved to be optimal both for the greedy algorithm and the FRiS-algorithm.

6 Conclusion

A method for the determination of the documents similarity was developed and tested based on comparing the attributes of bibliographic descriptions of the documents.

A study was also performed for various algorithms for documents clustering in order to identify the optimum algorithm for grouping the array of database records with the information about scientific publications into clusters containing articles devoted to similar subjects. The algorithms were tested on the database of the "Siberian mathematical journal" containing bibliographic descriptions of articles published from 2000 to 2005. The testing showed that the FRiS-algorithm is optimum for this task although the greedy algorithm yields fairly acceptable results too.

References

1. Salton, G.: Dynamic information and library processing. Prentice-Hall, Inc., Upper Saddle River (1975)
2. Voronin, Y.A.: Elements of Similarity Theory. Nauka, Novosibirsk (1991) (in Russian)
3. Cormen, T., Rivest, R., Leiserson, C.: Introduction to Algorithms. Massachusetts Institute of Technology (1990)
4. Zagoruiko, N.G., Borisova, I.A., Dyubanov, V.V., Kutnenko, O.A.: Methods of recognition based on the function of rival similarity. Pattern Recognition and Image Analysis 18(1), 1–6 (2008)
5. Barakhnin, V.B., Nekhaeva, V.A.: A technology for creation of object domain thesaurus using encyclopedic subject index. Computational Technolgies 12(Special issue), 3–9 (2007) (in Russian)

On the Problem of Prediction

Evgenii Vityaev[1,2] and Stanislav Smerdov[2]

[1] Sobolev Institute of Mathematics, Russian Academy of Sciences,
Koptyug prospect 4, Novosibirsk, 630090, Russia
[2] Novosibirsk State University
evgenii.vityaev@math.nsc.ru
http://www.math.nsc.ru/AP/ScientificDiscovery

Abstract. We consider predictions provided by Inductive-Statistical (I-S) inference. It was noted by Hempel that I-S inference is statistically ambiguous. To avoid this problem Hempel introduced the Requirement of Maximal Specificity (RMS). We define the formal notion of RMS in terms of probabilistic logic, and maximally specific rules (MS-rules), i. e. rules satisfying RMS. Then we prove that any set of MS-rules draws no contradictions in I-S inference, therefore predictions based on MS-rules avoid statistical ambiguity. I-S inference may be used for predictions in knowledge bases or expert systems. In the last we need to calculate the probabilistic estimations for predictions. Though one may use existing probabilistic logics or "quantitative deductions" to obtain these estimations, instead we define a semantic probabilistic inference and prove that it approximates logical inference in some sense. We also developed a program system 'Discovery' which realizes this inference and was successfully applied to the solution of many practical tasks.

Keywords: scientific discovery, probability and logic synthesis, probabilistic logic programming, machine learning.

1 Introduction

1.1 The Statistical Ambiguity Problem

One of the major results of the Philosophy of Science is the so-called *Covering Law Model*, which was introduced by Hempel in the early sixties in his famous article 'Aspects of Scientific Explanation' (see Hempel [1], [2], and Salmon [3] for a historical overview). The basic idea of this covering law model is that a fact is explained by subsumption under the so-called *covering law,* i.e. the task of an explanation is to show that a fact can be considered as an instantiation of a law. In the covering law model two types of explanation are distinguished: *Deductive-Nomological* explanations (D-N explanations) and *Inductive-Statistical* explanations (I-S explanations). In D-N explanations the laws are *deterministic*, whereas in I-S explanations the laws are *statistical.* Right from the beginning it was clear to Hempel that two I-S explanations can yield contradictory conclusions. He called this phenomenon the *statistical ambiguity* of I-S explanations [1], [2]. Let us consider the following example of the statistical ambiguity.

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 280–296, 2011.

Suppose that we have the following statements about Jane Jones. 'Almost all cases of streptococcus infection clear up quickly after the administration of penicillin'(L1). 'Almost no cases of penicillin resistant streptococcus infection clear up quickly after the administration of penicillin'(L2). 'Jane Jones had streptococcus infection'(C1). 'Jane Jones received treatment with penicillin'(C2). 'Jane Jones had a penicillin resistant streptococcus infection'(C3). From these statements it is possible to construct two contradictory arguments, one explaining why Jane Jones recovered quickly (E), and the other one, explaining its negation why Jane Jones did not recover quickly (¬E).

$$\begin{array}{cc} Argument1 & Argument2 \\ \dfrac{\dfrac{L1}{C1, C2}[r]}{E} & \dfrac{\dfrac{L2}{C2, C3}[r]}{\neg E} \end{array}$$

The premises of both arguments are consistent with each other, they could all be true. However, their conclusions contradict each other, making these arguments rival ones.

Hempel hoped to solve this problem by forcing all statistical laws in an argument to be maximally specific. That is, they should contain all relevant information with respect to the domain in question. In our example, then, premise C3 of the second argument invalidates the first argument, since the law L1 is not maximally specific with respect to all information about Jane Jones (presented treatment is intuitively clear, but not formal, because we don't have a precise definition of specificity yet – it will appear in the following sections). So, we can only explain ¬E, but not E.

1.2 Inductive-Statistical Inference

Hempel proposed the formalization of the statistical inference as Inductive-Statistical Inference (I-S inference) and the property of maximally specific statistical laws as the Requirement of Maximal Specificity (RMS). The Inductive-Statistical Inference has the form:

$$\frac{\dfrac{L_1, \ldots, L_{\mathrm{m}}}{C_1, \ldots, C_{\mathrm{n}}}[r]}{G}$$

It satisfies the following conditions:

- $L_1, \ldots, L_{\mathrm{m}}, C_1, \ldots, C_{\mathrm{n}} \vdash G$;
- $L_1, \ldots, L_{\mathrm{m}}, C_1, \ldots, C_{\mathrm{n}}$ are consistent;
- $L_1, \ldots, L_{\mathrm{m}} \nvdash G$; $C_1, \ldots, C_{\mathrm{n}} \nvdash G$;
- $L_1, \ldots, L_{\mathrm{m}}$ are composed of statistical quantified formulas.
- $C_1, \ldots, C_{\mathrm{n}}$ are quantifier-free;
- RMS: All laws $L_1, \ldots, L_{\mathrm{m}}$ are maximally specific.

In Hempel's [1], [2] the RMS is defined as follows. An I-S argument of the form:

$$\frac{\dfrac{p(G;F)}{F(\boldsymbol{a})}}{G(\boldsymbol{a})}[r]$$

is an acceptable I-S explanation with respect to a "knowledge state" K, if the following Requirement of Maximal Specificity is satisfied. For any class H for which the corresponding two sentences are contained in K

$$\begin{aligned} &\forall x(H(x) \Rightarrow F(x)), \\ &H(\boldsymbol{a}), \end{aligned} \tag{1}$$

there exists a statistical law p(G;H) = r' in K such that r = r'. The basic idea of RMS is that if F and H both contain the object $\boldsymbol{a}$, and H is a subset of F, then H provides more specific information about the object $\boldsymbol{a}$ than F, and therefore the law p(G;H) should be preferred over the law p(G; F).

1.3 The Requirement of Maximal Specificity in Default Logic

Nowadays the same problems arise in non-monotonic logic and especially in default logic. Hempel's RMS produces also non-monotonic effects in inductive statistical reasoning. The streptococcus infection example is non-monotonic in the following sense. It was observed that the conflict between argument 1 and the argument 2 depends on the knowledge state K. If K contains only the information that John is infected, then RMS determines that argument 1 is the best (or the most specific) explanation: since no additional information (such as C3) is given, so L1 is maximally specific according to K. In that case, K implies the conclusion that John will recover quickly. However, if K is expanded with the premise C3, i.e. the information that John had a penicillin resistant streptococcus infection, then RMS determines that argument 2 explains that John will not recover quickly. Hence, the conclusion that John will recover quickly is not preserved under expansion of K.

Yao-Hua Tan [4] showed that there is a remarkable resemblance between two research traditions: default logic and inductive-statistical explanations. Both research traditions have the same research objective; to develop formalisms for reasoning with incomplete information. In both research traditions the crucial problem that had to be dealt with is the problem of *Specificity*, i.e. when two arguments conflict with each other the most specific argument has to be preferred to the less specific argument. This criterion of specificity, that was proposed in AI research, is very similar to the criterion of maximal specificity suggested by Hempel in the early sixties.

Let us formulate the Requirement of Maximal Specificity (RMS*) in default logic. Essentially, default logic is an ordinary first-order predicate logic extended with extra inference rules that are called default rules. The logical form of a *default rule* is:

$$(\alpha(x) : \beta_1(x), \ldots, \beta_{\mathrm{n}}(x)/\omega(x))$$

The subformulas $\alpha(x)$, $\beta_i(x)$, and $\omega(x)$ are predicate logical formulas with free variable x. The subformula $\alpha(x)$ is called the *prerequisite*, $\beta_i(x)$ are the *justifications* and $\omega(x)$ is the *consequent* of the default rule. The intuitive interpretation of a default rule follows: if the prerequisite $\alpha(x)$ is valid, and all justifications $\beta_i(x)$ are consistent with the available information (i.e. $\neg\beta_i(x)$ is not derivable from the available information), then one can assume that the consequent $\omega(x)$ is valid.

A set of formulas E is an *extension* of the default theory $\Delta = \langle W;D\rangle$, D – the set of default rules, W – a set of predicate logical formulas, if E is the smallest set such as: $W \subseteq E$; $E = Th(E)$; for each default rule $(\alpha(x):\beta_1(x),\ldots,\beta_n(x)/\omega(x)) \in D$, and each term t: if $\alpha(t) \in E$, and $\neg\beta_1(t),\ldots,\neg\beta_n(t) \notin E$, then $\omega(t) \in E$.

RMS*: If a default theory has multiple conflicting extensions, then the extension is preferred which is generated by the most specific defaults [4].

The default rule with the 'most specific' prerequisite is preferred in case of conflicts. Let A(x) and B(x) be the prerequisites of the default rules D1 and D2. The prerequisite A(x) is *more specific* than B(x) if the set that the predicate A refers to is a subset of the set that B refers to, i.e. if the sentence $\forall x(A(x) \Rightarrow B(x))$ is valid. It is obvious that this criterion can be considered as the analogue of RMS in default logic.

1.4 The Solution of the Statistical Ambiguity Problem

From the previous consideration we see that the statistical ambiguity problem raises in AI in different forms, but it isn't solved hitherto. We will once again state the problem:

- is it possible to define the RMS in such a way that it solves the statistical ambiguity problem?
- can we define the RMS in such a way that the set of sentences satisfying the RMS will be consistent?

This problem is very important, because it means the consistency of predictions. The predictions nowadays are produced by different AI systems. In this paper we present our solution of this problem. We define the set of Maximally Specific Rules (MSR) and the Requirement of Maximal Specificity (RMS) and prove that sentences from MSR satisfy RMS and that the set MSR is consistent.

1.5 Probabilistic Approximation of Empirical Theories

Let us consider the task of the empirical theory discovery in the presence of noise provided for example the propensity interpretation of probability by Karl Popper.

Let $\mathfrak{L}$ be the first-order logic with signature $\Im = \langle P_1, ..., P_m\rangle$, $m > 0$, where $P_1, ..., P_m$ are predicate symbols of arity n_1,..., n_m, and fixed tuples of variables attached to each predicate symbol (so every predicate appears only with its own variables — this situation is quite similar to propositional classical logic). An empirical system [5] is taken to mean a finite model $\mathfrak{M} = \langle B,W\rangle$ of the signature $\Im$, where B is the basic set of the empirical system, and $W = \langle P_1,...,P_m\rangle$ is the

tuple of predicates of the signature $\Im$ defined on B. Let $\mathrm{Th}(\mathfrak{M})$ be the set of all rules, which are true on the empirical system $\mathfrak{M}$ and having the form:

$$C = (A_1 \& ... \& A_{\mathrm{k}} \Rightarrow A_0),\ k \geq 0 \tag{2}$$

where $A_0, A_1, ..., A_k$ are literals.

In the next section we define the notion of law and the set of all laws $\mathcal{L}$ and prove that $\mathcal{L} \vdash \mathrm{Th}(\mathfrak{M})$. Hence, we can solve the task of empirical theory discovery by discovering all laws from the set $\mathcal{L}$. In section 5 we prove that $\mathcal{L} \subset$ MSR and in section 7 we prove that MSR is consistent. Therefore MSR $\vdash \mathrm{Th}(\mathfrak{M})$ and MSR provides a probabilistic approximation of the empirical theory $\mathrm{Th}(\mathfrak{M})$. See the review of Jon Williamson [6] for other approaches.

1.6 Approximation of Logical Inference by Semantic Probabilistic Inference

So far we considered I-S-inferences using one rule for the inference. In general, I-S-inference uses many rules for the inference in knowledge bases and expert systems. This inference is based on the logical inference rules. The probability estimations of the inference results are obtained by the probabilistic logics or so called "quantitative deductions" [10], [11]. These estimations not always produce satisfactory results.

We replace the logical inference by the special semantic probabilistic inference, which produces all rules of the sets L, MSR, and also approximates the logical inference. We prove (Theorem 7) that estimations produced by the semantic probabilistic inference are no less (or even greater) than estimations produced by the probabilistic logics based on the logical inference.

2 Laws

Proposition 1. *The rule* $C = (A_1 \& ... \& A_{\mathrm{k}} \Rightarrow A_0)$ *logically follows from any rule of the form:*

$$(B_1 \& ... \& B_{\mathrm{h}} \Rightarrow A_0) \vdash (A_1 \& ... \& A_{\mathrm{k}} \Rightarrow A_0), \tag{3}$$
$$\{B_1, ..., B_{\mathrm{h}}\} \subset \{A_1, ..., A_{\mathrm{k}}\},\ 0 \leq h < k$$

Definition 1. *By a subrule of the rule* $C = (A_1 \& ... \& A_{\mathrm{k}} \Rightarrow A_0)$ *we mean any logically stronger rule of the form (3).*

Corollary 1. *If a subrule of the rule C is true on* $\mathfrak{M}$*, then the rule C is also true on* $\mathfrak{M}$*.*

Definition 2. *By a law on* $\mathfrak{M}$*, we mean any rule C of the form (2) that satisfies the following conditions [7], [8]:*

(1) C is true on $\mathfrak{M}$*;*
(2) the premise of the rule is not always false on $\mathfrak{M}$*;*
(3) none of its subrules is true on $\mathfrak{M}$*.*

Let $\mathcal{L}$ be the set of all laws on $\mathfrak{M}$.

Theorem 1. [7]. $\mathcal{L} \vdash \mathrm{Th}(\mathfrak{M})$.

3 The Probability of Events and Sentences

Let us generalize the notion of law into the probabilistic case. For this purpose we introduce the probability on the model $\mathfrak{M}$. For the sake of simplicity we define the probability in the most simple case (follow the paper [9]). More general definitions of probability function μ are considered in [9]. Further considerations don't depend on the selected probability definition and, for example, true for definition 10 below.

We introduce the probability μ as a discrete function on B (B should be countable), μ: B $\rightarrow$ [0,1] such that

$$\sum_{a \in B} \mu(a) = 1, \; and \; \mu(a) \neq 0, \; a \in B; \; \mu(D) = \sum_{b \in D} \mu(b), \; D \subseteq B \qquad (4)$$

We define the probability μ on the product B^n as a probability function $\mu^n(a_1, ..., a_n) = \mu(a_1) \times ... \times \mu(a_n)$.

Let us define the interpretation of the language $\mathfrak{L}$ on the empirical system $\mathfrak{M}$ = $\langle$B,W$\rangle$ as mapping I: $\Im \rightarrow$ W, which associates with every signature symbol $P_j \in \Im$, j = 1,...,m, the predicate P_j from W of the same arity. Let X = {x_1, x_2, x_3, ... } be the set of all variables of the language $\mathfrak{L}$. By the validation ν is meant the function ν: X $\rightarrow$ B, mapping variables into the set of objects B.

Let us define the probability for the sentences of the language $\mathfrak{L}$. Let U($\Im$) be the set of all atomic formulas of the language $\mathfrak{L}$; $\Re(\Im)$ is the set of all sentences of the language $\mathfrak{L}$, obtained by closure of the set U($\Im$) with respect to standard Boolean operations &, $\vee$, $\neg$. By the $\hat{\varphi}$, $\varphi \in \Re(\Im)$ we define the formula, where the predicate symbols of $\Im$ are substituted by the predicates of W via interpretation I and by the $\nu\hat{\varphi}$ we define the formula, where variables of the formula $\hat{\varphi}$ are substituted by the objects of A via the validation ν. In particular, $\nu\hat{P}_j(x_1^j,...,x_{n_j}^j)^{\varepsilon j}$ = $P_j(a_1,...,a_j)^{\varepsilon j}$, $\nu(x_1^j) = a_1, \ldots, \nu(x_{n_j}^j) = a_j$. Let us define the probability η of the sentences of $\Re(\Im)$. If $x_1,\ldots,x_n$ are all variables of the sentence $\varphi \in \Re(\Im)$, then

$$\eta(\varphi) = \mu^n(\{(a_1, \ldots, a_n) \,|\, \nu\hat{\varphi} \; is \; true \; on \; \mathfrak{M}, \nu(x_1) = a_1, \ldots, \nu(x_n) = a_n\}) \qquad (5)$$

4 The Probabilistic Laws on $\mathfrak{M}$

Let us revise the concept of the law on $\mathfrak{M}$ in terms of probability. We do it in such a way that the concept of the law on $\mathfrak{M}$ would be a particular case of this definition. A law on $\mathfrak{M}$ is a true rule such that all its subrules are false on $\mathfrak{M}$ or, in other words, a law is such a true rule that cannot be made simpler or logically stronger without losing its truth. This property of the law "not to be simplified" allows stating the law not only in terms of truth but also in terms of probability.

For the rule C = (A_1& ...&$A_k \Rightarrow A_0$) we define the conditional probability as η(C) = $\eta(A_0/A_1\& \ldots\&A_k) = \eta(A_0\&A_1\& \ldots\&A_k)/\eta(A_1\& \ldots\&A_k)$.

Theorem 2. [7]. For any rule C = ($A_1\&\ldots\&A_k \Rightarrow A_0$), the following two conditions are equivalent:

1. a rule C is a law on $\mathfrak{M}$, that is, it satisfies properties (1–3) of Definition 2;
2. (a) $\eta(\mathrm{C}) = 1$;
   (b) $\eta(\mathrm{A_1\&...\&A_k}) > 0$;
   (c) the conditional probability $\eta(\mathrm{C})$ of the rule C is strictly greater than the conditional probability of each of its subrules.

This theorem gives us the equivalent definition of the law on $\mathfrak{M}$.

Definition 3. *By a probabilistic law on $\mathfrak{M}$ with conditional probability 1 is meant the rule $C = (A_1\&...\&A_k \Rightarrow A_0)$ of the form (2) satisfying the following conditions:*

1. *$\eta(C) = 1$, $\eta(A_1\&...\&A_k) > 0$;*
2. *conditional probability of the rule $\eta(C)$ is strictly greater than the conditional probability of each of its subrules.*

The next corollary follows from Theorem 2.

Corollary 2. *A rule is a probabilistic law on $\mathfrak{M}$ with conditional probability 1 iff it is a law on $\mathfrak{M}$.*

Let us consider items 1 and 2 of Theorem 2 from the standpoint of the 'not to be simplified' law:

- A law is such a true on $\mathfrak{M}$ rule that cannot be simplified or made logically stronger without loss of truth.
- Any logically stronger subrule of a rule has has a conditional probability smaller than 1, so the rule cannot be simplified without loosing the value 1 of the conditional probability.

A more general definition of a law follows:

Definition 4. *A law is such a rule of the form (2) based on truth values, conditional probability or other evaluations of the sentences, which cannot be made logically stronger without reducing their values.*

Therefore, we can define the probabilistic law for the more general case by omitting the condition $\eta(\mathrm{C}) = 1$ from the point (1) of Definition 3.

Definition 5. *By a probabilistic law on $\mathfrak{M}$ we mean such a rule $C = (A_1\&...\&A_k \Rightarrow A_0)$ of the form (2), the conditional probability of which is defined and strictly greater than the conditional probability of each of its subrules. In particular, the conditional probability $\eta(C)$ of the rule C is strictly greater than the probability $\eta(A_0)$, which is the probability of the subrule ($\Rightarrow A_0$).*

Let us denote by $\mathcal{LP}$ the set of all probabilistic laws. It follows from Theorem 2 and Definition 5 that the set $\mathcal{LP}$ includes the set $\mathcal{L}$.

Corollary 3. $\mathcal{L} \subseteq \mathcal{LP}$.

Definition 6. *By a Strongest Probabilistic Law (SPL-rule) on $\mathfrak{M}$, we designate such a probabilistic law $C = (A_1\& ...\&A_k \Rightarrow A_0)$, which is not a subrule of any other probabilistic law.*

We define as SPL the set of all SPL-rules.

Proposition 2. $\mathcal{L} \subseteq SPL \subseteq \mathcal{LP}$.

5 Semantic Probabilistic Inference

Let us define the Semantic Probabilistic inference (SP-inference) of the sets of laws $\mathcal{L}$ and probabilistic laws $\mathcal{LP}$.

Definition 7. [7], [8], [16]. By a *Semantic Probabilistic inference* of some SPL-rule C we mean a sequence $C_1, C_2, \ldots, C_{\mathrm{n}} = C \in \mathcal{LP}$, denoted by $C_1 \sqsubset C_2 \sqsubset \cdots \sqsubset C_{\mathrm{n}}$, such that:

$$C_{\mathrm{i}} = (A_1^i \& \ldots \& A_{k_i}^i \Rightarrow G),\ i = 1, 2, \ldots n,\ n > 0, \qquad (6)$$
$$\textit{the rules } C_{\mathrm{i}} \textit{ are subrules of the rules } C_{\mathrm{i+1}},$$
$$\eta(C_{\mathrm{i+1}}) > \eta(C_{\mathrm{i}}),\ i = 1, 2, \ldots, n-1,$$

and this sequence is maximal, i.e., there is no $C' \in \mathcal{LP}$ such that $\eta(C') > \eta(C)$ and C is a subrule of C'.

Unlike probabilistic logics [10], [11] the probability of the sentences is strictly increase in the process of SP-inference.

Proposition 3. *Any probabilistic law from $\mathcal{LP}$ belongs to some SP-inference. For any SPL-rule there is some SP-inference of that rule.*

Corollary 4. *For any law from $\mathcal{L}$ there is some SP-inference of that law.*

Let us consider the set of all SP-inferences of the sentence G. This set constitutes the Semantic Probabilistic Inference lattice (SPI-lattice) of this sentence.

Definition 8. *By a maximally specific rule MS(G) of a sentence G we mean a SPL-rule of the SPI-lattice of the sentence G, which has the maximum value of conditional probability among all other SPL-rules of the SPI-lattice.*

We define as MSR the set of all maximally specific rules.

Proposition 4. $\mathcal{L} \subseteq MSR \subseteq SPL \subseteq \mathcal{LP}$

6 Probabilistic Maximally Specific Laws

Now we define the Requirement of Maximal Specificity (RMS). We will suppose that the class H of objects in (1) is defined by some sentence $\mathrm{H} \in \Re(\Im)$ of the language $\mathfrak{L}$. In this case the RMS says that $\mathrm{p(G;H)} = \mathrm{p(G;F)} = \mathrm{r}$ for this sentence. In terms of probability η it means that $\eta(\mathrm{G/H}) = \eta(\mathrm{G/F}) = \mathrm{r}$ for any $\mathrm{H} \in \Re(\Im)$ satisfying (1).

Definition 9. *The Requirement of Maximal Specificity (RMS):*
if we add any sentence $H \in \Re(\Im)$ to the premise of the rule $(F \Rightarrow G)$, $\eta(G/F) = r$, such that $F(\mathbf{a})\&H(\mathbf{a})$ for some object $\mathbf{a}$, then for the new rule $(F\&H \Rightarrow G)$ we have $\eta(G/F\&H) = \eta(G/F) = r$.

In other words, the requirement RMS means that there is no other sentence H in $\Re(\Im)$ that increases (or decreases, see lemma 1 below) the conditional probability $\eta(G/F) = r$ by adding it to the premise.

Lemma 1. [8]. If the sentence $H \in \Re(\Im)$ decreases the probability $\eta(G/F\&H) < \eta(G/F)$ then the sentence $\neg H$ increases it: $\eta(G/F\&\neg H) > \eta(G/F)$.

Lemma 2. [8]. For any rule $C = (B_1\&...\&B_t \Rightarrow A_0)$, $\eta(B_1\&...\&B_t) > 0$, of the form (2) there is a probabilistic law $C' = (A_1\&...\&A_k \Rightarrow A_0)$ on $\mathfrak{M}$ which is a subrule of the rule C and $\eta(C') \geq \eta(C)$.

Theorem 3. [8]. Any MS(G) rule satisfies the RMS requirement.

Corollary 5. [8]. Any law on $\mathfrak{M}$ satisfies the RMS requirement.

7 The Solution of the Statistical Ambiguity Problem

Theorem 4. [8]. The I-S inference is consistent for any theory $Th \subseteq MSR$ in the following sense: it is impossible to obtain a contradiction (ambiguity) in I-S inference using only rules from Th, i. e. there are no $(A \Rightarrow G) \in Th$ and $(B \Rightarrow \neg G) \in Th$ such that $\eta(A\&B) > 0$.

Let us illustrate this theorem by the example of Jane Jones. We can define the maximally specific rules MS(E), MS($\neg$E) for the sentences E, $\neg$E as follows:

$\widehat{L1}$: 'Almost all cases of streptococcus infection, that are not resistant to streptococcus infection, clear up quickly after the administration of penicillin';
L2 : 'Almost no cases of penicillin resistant streptococcus infection clear up quickly after the administration of penicillin'.

The rule $\widehat{L1}$ has a greater value of conditional probability than the rule L1; hence, it is a MS(E) rule for the sentence E. These two rules can't be fulfilled on the same data.

So, we can predict without contradictions, if we use the set MSR as statistical laws in I-S inference.

8 Probabilistic Herbrand Models

In the following, we perform all considerations in the frame of logical programming with functional symbols, substitutions, and the countable set of variables. For that purpose we extend our language and redefine some definitions. Next sections 8-12 are updated and translated version of the paper [16].

Consider a first order language L with equality of the finite signature $\Omega = \langle P_1, P_2, ..., P_{n_1}, f_1, f_2, ..., f_{n_2}, c_1, c_2, ..., c_{n_3}\rangle$. Let U denotes a set of all *ground terms* (without free variables), X – a countable set of *variables*, T – a set of *terms*, F_L – a set of *formulas*, F – a set of *formulas without quantifiers*, S – a set

of *sentences* (formulas without free variables), $\Re = \mathrm{F} \cap \mathrm{S}$ – the set of all *ground sentences* of the signature, B_L – the set of all ground atoms of the signature Ω.

A mapping θ: X $\rightarrow$ T is called a *substitution*. Denote by Θ the set of all substitutions. The substitution $\theta(\mathrm{x}) = \mathrm{x}$ is called identical. Substitutions are naturally extended to arbitrarily expressions. Thus, substitutions for the term $\mathrm{t} = \mathrm{f}(\mathrm{t}_1,...,\mathrm{t}_\mathrm{n})$ and atom $\mathrm{A} = \mathrm{P}(\mathrm{t}_1,...,\mathrm{t}_\mathrm{n})$ are equal to $\theta\mathrm{t} = \mathrm{f}(\theta\mathrm{t}_1,...,\theta\mathrm{t}_\mathrm{n})$, $\theta\mathrm{A} = \mathrm{P}(\theta\mathrm{t}_1,...,\theta\mathrm{t}_\mathrm{n})$ respectively. A rule $\theta\mathrm{A} \leftarrow \theta\mathrm{A}_1,...,\theta\mathrm{A}_\mathrm{n}$ is a variant of the rule $\mathrm{A} \leftarrow \mathrm{A}_1,\ldots,\mathrm{A}_\mathrm{n}$ if θ is a permutation of the set X.

Following [9], let us define a probability μ on a subset F' $\subseteq$ F, F' $\neq \emptyset$ of sentences closed with respect to logical operations & ,v,$\neg$ and term substitutions.

Definition 10. *A mapping μ: F' $\rightarrow$ [0,1] is called a probability provided that the following conditions are satisfied:*

1) if $\vdash \phi$, then $\mu(\phi) = 1$;

2) if $\vdash \neg(\phi\&\psi)$, then $\mu(\phi \vee \psi) = \mu(\phi) + \mu(\psi)$;

Definition 11. *A pair $M = \langle U, \mu \rangle$, where μ is a probability on $\Re$, is called a probabilistic Herbrand model of signature Ω.*

Definition 12. *A pair $M = \langle U, \mu \rangle$, where $I : B_L \rightarrow \{0, 1\}$, is called a Herbrand model of signature Ω.*

Let there be given a certain class $G \subseteq 2^{B_L}$ of Herbrand models (a set of possible worlds) and a probability μ on F. For every $\varphi \in$F let $G(\phi) = \{M | M \in G, M \models \phi\}$, where $\models$ denotes the satisfaction, and let D= $\{G(\phi) \mid \phi \in \mathrm{F}\}$.

Definition 13. *A class G of Herbrand models is said to be coordinated with the probability μ on the set of formulas F', and a probabilistic Herbrand model $M = \langle U, \mu \rangle$ is said to be a probabilistic model of the class G if $\mu(\phi) = 0$ follows from $G(\phi) = \emptyset$, $\phi \in F$.*

We will consider two cases: $F' = F$ and $F' = \Re$. In the first case, the probability expands on sentences with free variables as $\mu(\varphi) = \inf\limits_{\theta \in \Theta G} \{\mu(\varphi\theta)\}$, where ΘG is the set of all substitutions of variables by ground terms.

9 Logical Programs

Let PR denotes a set of all *rules* $A \leftarrow A_1, ..., A_k$, $k \geq 0$ of the signature Ω, where $A, A_1, ..., A_k$ are atoms of the signature Ω. If atom A is absent, then the rule $\leftarrow A_1, ..., A_k$ $(k \geqslant 0)$ is called a *goal* (*request*). In requests we will write '&' between atoms instead of ','. If $k = 0$, then the rule A $\leftarrow$ is called a *fact*. A *logic program* Pr is a finite collection of rules.

Let fix a *selection rule* R, which selects one of the atoms from a request. Let $N = \leftarrow A_1\&...\&A_i\&...\&A_k$, $k \geq 1$ be a request, where the rule R selects the atom A_i, and a rule $C = A \leftarrow B_1, ..., B_m$ be a variant of some rule of the program

Pr, where all the variables are different from those of the request. Let θ be the most general unification of the atoms A_i, A ($A_i\theta = A\theta$). Then the requests

$$\leftarrow (A_1\&...\&A_{i-1}\&B_1\&...\&B_m\&A_{i+1}\&...\&A_k)\theta, if m \geq 1 \qquad (7)$$
$$\leftarrow (A_1\&...\&A_i\&...\&A_k)\theta, if m = 0$$

are called *inferred* from the request N by the rule $C = A \leftarrow B_1, ..., B_m$ with the help of the substitution θ and the selection rule R. It is seen from the definition, that the atom A_i is not removed from the request after the unification with a certain program fact. Such atoms will be underlined. Suppose, that the rule R does not select the underlined atoms for the next inference steps.

The set of all possible requests of the signature Ω with the given relation of inference is called a *calculation space of the program Pr and the selection rule R.* A maximal sequence of requests $N = N_0, N_1, N_2, ...$ together with the sequence of rules $C_0, C_1, C_2, ...$ and unification's $\theta_0, \theta_1, \theta_2, ...$ such, that requests N_{i+1} are inferred from the requests N_i by means of the rules C_i, substitutions θ_i and the selection rule R, i = 1,2,... is called a *SLDF-inference* (Linear resolution with Selection rule for Definite clauses and underlined Facts) of the goal N in the calculation space. A SLDF-inference is a maximal path in the calculation space, starting with N. A SLDF-inference ending by a request with all atoms underlined is called *successful*. A finite inference, which is not successful, is called *dead ended*. A set of all SLDF-inferences starting with the goal N can be presented in the form of a tree (a prefix tree of SLDF-inferences). This tree is called the *SLDF-tree of the request N calculations*. The SLDF-tree containing a successful SLDF-inference is called a *successful SLDF-tree*.

10 Estimations of the Probability and Conditional Probability of Requests

Let $M = \langle U, \mu \rangle$ be a probabilistic Herbrand model. Consider a successful SLDF-inference $N = N_0, N_1, ..., N_k$ of the request N in a calculation space of the program Pr obtained by means of the sequence of rules $C_0, C_1, ..., C_{k-1}$, unification's $\theta_0, \theta_1, ..., \theta_{k-1}$, $\theta \leftrightharpoons \theta_0\theta_1...\theta_{k-1}$ and the selection rule R (here we suppose that any N_i and N_j have no common variables – this modification can be easily performed by choosing the appropriate variants of rules $C_0, \ldots, C_{k-1}$).

It is not difficult to show that the sequence of requests $N\theta, N_1\theta, ..., N_k\theta = N_k$ is also a successful SLDF-inference of the request $N\theta$ by means of the same sequence of rules $C_0\theta, C_1\theta, ..., C_{k-1}\theta$, identical unifications and the selection rule R.

The probability of a rule $C = A \leftarrow B_1, ..., B_m$ ($m \geq 1$) is defined and equal to

$$\mu(C) = \mu(A|B_1\&...\&B_m) = \mu(A\&B_1\&...\&B_m)/\mu(B_1\&...\&B_m)$$

iff $\mu(B_1\&...\&B_m) \neq 0$ and it is undefined otherwise. Represent facts $A \leftarrow$ by the rules $A \leftarrow true$. Then $\mu(C) = \mu(A|true) = \mu(A)$. Suppose that $\mu(C)$ means that the probability is defined. Denote by $PR_0 \subseteq PR$ the set of all rules, for which conditional probability μ is defined; $Pr_0 \leftrightharpoons PR_0 \cap \Pr$.

Definition 14. *A rule C is true on a Herbrand model* $N \in 2^{B_L}$ *(*$N \models C$*) iff it is true on N under any state (for any mapping* ρ*:* $X \to U$*).*

Definition 15. *A program Pr is true on a Herbrand model N (*$N \models \mathrm{Pr}$*) iff each rule of the program is true on N.*

Definition 16. *A program Pr is true on a class G of models iff* $\forall N \in G$, $N \models \mathrm{Pr}$.

We will write $C \in F'$ for the rule $C \leftrightharpoons A \leftarrow B_1 \& ... \& B_m$, as soon as $A, B_1, ..., B_m \in F'$.

Proposition 5. *If* $C \in \mathrm{Pr} \cap F'$, $C = A \leftarrow B_1, ..., B_m$, $\mu(B_1 \& ... \& B_m) > 0$, *then*

$$\mu(\neg(B_1 \& ... \& B_m) \vee A) = 1 \Leftrightarrow \mu(C) = 1.$$

Corollary 6. *If the program Pr is true on the class of Herbrand models G, which is coordinated with the probability* μ *on the set of formulas F', then* $\mu(C) = 1$, $C \in \mathrm{Pr} \cap F'$, *if it is defined.*

Denote the conjunction of all non-underlined atoms of the request N_i by $N_i^{\wedge}$. If all atoms are underlined (as in the request $\mathrm{N_k}$), then $N_k^{\wedge} = true$. Denote the conjunction of all underlined atoms of the request N_i by $N_i F^{\wedge}$. Then, $N_i F^{\wedge}$ is a conjunction of all facts used in the SLDF-inference of the request $N\theta$.

Consider the inference of the requests (7) from the request

$$N\theta = \leftarrow (A_1 \& ... \& A_i \& ... \& A_k)\theta,\ k \geq 1$$

by means of the rule $C = A \leftarrow B_1, ..., B_m$. Let us estimate the probabilities $\mu(N\theta^{\wedge})$, $\mu(N\theta^{\wedge}|N_k F^{\wedge})$ assuming, that only probabilities $\mu(N_1\theta^{\wedge})$, $\mu(A_i\theta)$, $\mu(B\theta)$ and $p = \mu(A\theta|B\theta^{\wedge})$ are known, where B stands for the conjunction $B_1 \wedge \cdots \wedge B_m$.

Lemma 3. [16]. If $\mu(N_1\theta^{\wedge}) > 0$ and $\mu(B\theta) > 0$, then:
1) $\mu(N\theta^{\wedge}) \leq \mu(\neg B\theta^{\wedge}) + \min\{\mu(N_1\theta^{\wedge}), \mu(A\theta \& B\theta^{\wedge})\}$;
2) $\mu(N\theta^{\wedge}) \geq \mu(N_1\theta^{\wedge}) - (1-p)\mu(B\theta^{\wedge})$;
3) $\mu(N\theta^{\wedge}|N_1\theta^{\wedge}) \leq p/\mu(N\theta^{\wedge}|B\theta^{\wedge})$;
4) $\mu(N\theta^{\wedge}|N_1\theta^{\wedge}) \geq 1 - (1-p)/\mu(N\theta^{\wedge}|B\theta^{\wedge})$.

Corollary 7. [16]. If $\mu(N_1\theta^{\wedge}) > 0$, $\mu(B\theta) > 0$ and $p = 1$, then:
1) $\mu(N_1\theta^{\wedge}) \leq \mu(N\theta^{\wedge}) \leq \min\{1, \mu(\neg B\theta^{\wedge}) + \mu(N_1\theta^{\wedge})\}$;
2) $\mu(N\theta^{\wedge}|N_1\theta^{\wedge}) = 1$.

Corollary 8. [16]. If $\mu(N_1\theta^{\wedge}) > 0$ and the rule is the fact $(A \leftarrow true)\theta$, $\mu(B\theta) = 1$, then:
1) $\mu(N\theta^{\wedge}) \leq \min\{\mu(N_1\theta^{\wedge}), \mu(A\theta)\}$;
2) $\mu(N\theta^{\wedge}) \geq \mu(N_1\theta^{\wedge}) + \mu(A\theta) - 1$;
3) $\mu(N\theta^{\wedge}|N_1\theta^{\wedge}) \leq \mu(A\theta)/\mu(N_1\theta^{\wedge})$;
4) $\mu(N\theta^{\wedge}|N_1\theta^{\wedge}) \geq 1 - (1-\mu(A\theta))/\mu(N_1\theta^{\wedge})$.

Corollary 9. [16]. If $\mu(B\theta) > 0$, then:
1) $\mu(N\theta^{\wedge}\&B\theta^{\wedge}) \le \min\{\mu(N_1\theta^{\wedge}), \mu(A\theta\&B\theta^{\wedge})\}$;
2) $\mu(N\theta^{\wedge}\&B\theta^{\wedge}) \ge \mu(N_1\theta^{\wedge}) - (1-p)\mu(B\theta^{\wedge})$.

Consider the SLDF-inference $N\theta, N_1\theta, ..., N_k$ of the request Nθ by means of sequence of rules $C_i\theta = (A^i \leftarrow B^i_1, ..., B^i_{k_i})\theta$, $i = 0, ..., k-1$ and empty unifications. Denote $B^i\theta = (B^i_1\&...\&B^i_{k_i})\theta$, $p_i = \mu(C_i\theta)$.

Theorem 5. [16]. If $\mu(B^i\theta) > 0$, $i = 0, ..., k-1$, then under the previous conditions

$$\mu(N\theta^{\wedge}\&A^0\theta\&...\&A^{k-1}\theta) \ge 1 - \sum_{i=0}^{k-1}(1-p_i)\mu(B^i\theta)$$

Corollary 10. [16]. If $\mu(B^i\theta) > 0$, $i = 0, ..., k-1$, then under the previous conditions

$$\mu(N\theta^{\wedge}) \ge 1 - \sum_{i=0}^{k-1}(1-p_i)\mu(B^i\theta).$$

For every successful SLDF-inference $N\theta = N_0\theta, N_1\theta, ..., N_{k-1}\theta, N_k$, there exists a SLDF$'$-inference $N\theta = N'_0\theta, N'_1\theta, ..., N'_i\theta, , ..., N'_{k-1}, N'_k = N_k$, where facts are used in the last turn and the rules $C_j\theta$ with $k_j \ge 1$; $j = 1, ..., i-1$ are applied before facts. Then the request $N'_i\theta$ has the form $\leftarrow A'_1, ..., A'_s$, and the request N_k - the form $\leftarrow A'_1, ..., A'_s$. Such a SLDF$'$-inference is called *normalized*.

Theorem 6. [16]. If $\mu(B^j\theta) > 0$, j = 0,1, ..., i-1, and $\mu(N_kF^{\wedge}) > 0$, then for a successful SLDF-inference as defined earlier

$$\mu(N\theta^{\wedge}|N_kF^{\wedge}) \ge 1 - \sum_{j=0}^{i-1}(1-p_j)\mu(B^j\theta)/\mu(N_kF^{\wedge}),$$

where p_j – conditional probabilities, $B^j\theta$ – conditions of the rules C_j, $j = 1, ..., i-1$.

Let us define the probability estimations $\nu(N)$, $\eta(N)$ of the calculation space requests for the program Pr and selection rule R. Consider the SLDF-tree of some request N in the calculation space. If the SLDF-tree is not successful, then estimations ν and η are not defined. For the successful SLDF-tree consider a set $\{SLDF'_i\}_{i\in I}$, $I \ne \emptyset$ of all successful normalized SLDF-inferences of the requests $\{N\theta^i\}_{i\in I}$.

Determine the estimations $\{\nu^i\}_{i\in I}$, which are equal to the right-hand side of the inequality of Corollary 10, for the probabilities $\mu(N\theta^{i\wedge}) \ge \nu^i$ $(\forall i \in I)$ of the requests $\{N\theta^i\}_{i\in I}$ obtained by corresponding inferences. Determine also the estimates $\{\eta^i\}_{i\in I}$, which are equal to the right-hand side of the inequality of the theorem for the conditional probabilities $\mu(N\theta^{i\wedge}|N^i_{ki}F^{\wedge}) \ge \eta^i(\forall i \in I)$ of the requests $\{N\theta^i\}_{i\in I}$. Define $\nu(N) = \sup\limits_{i\in I}\{\nu^i\}$, $\eta(N) = \sup\limits_{i\in I}\{\eta^i\}$.

The SLDF-inference of the request N, where the estimation η(N) is reached, is called a *prediction of the request N*. The value η(N) is called the *estimation of the request N prediction*. If the prediction is not defined, then the estimation of the prediction η(N) is not defined.

Define the *relation* $\triangleright$ - "*to be more common*" on the set PR. Denote the set of all substitutions, which are not rearrangements, by Θt (the identical substitution belongs to Θt).

Definition 17. *The relation* $C' \triangleright C$, $C = A \leftarrow B_1, ..., B_m, C' = A' \leftarrow B'_1, ..., B'_{m'}$, $m, m' \geq 0$ *takes place, iff there exists a substitution* $\theta \in \Theta$ *such, that* $A'\theta = A, \{B'_1\theta, ..., B'_{m'}\theta\} \subseteq \{B_1, ..., B_m\}$ *and either* $\theta \in \Theta t$ *is not an identical substitution or* $m' < m$.

11 Inductive Synthesis of Probabilistic Logic Programs

A full set of facts for the class of models G represents a collection of sets $F(N) = \{A \leftarrow \mid N \models A$ for any state of the atom $A\}$, $N \in G$. Any finite collection D of finite subsets D(N) $\subset$ F(N) is called *data*. A probabilistic Herbrand model M = $\langle U, \mu \rangle$, which is in accordance with the class of models G, is called a *probabilistic Herbrand model of data* D.

In what way should the rules $C = A \leftarrow B_1, ..., B_m$, $m \geq 1$ be used for predictions? If, at a certain substitution, $\theta \in \Theta$ the conjunction $(B_1 \& ... \& B_m)\theta$ is true on a certain model N, which was chosen randomly from G in accordance with the measure μ, i.e. $\{B_1\theta, ..., B_m\theta\} \subseteq F(N)$, then the conclusion $A\theta$ is true on N with the probability $\mu(A\theta|(B_1 \& ... \& B_m)\theta) \geq \mu(A|B_1 \& ... \& B_m) = \mu(C)$. Thus, the probability μ(C) for rules with variables gives a lower bound for prediction probabilities of the atom $A\theta$. Note that only one model N, chosen arbitrarily from G, and the corresponding data D(N) should be used for predictions.

Definition 18. *The relation* $C' \sqsubset C \leftrightharpoons (C' \triangleright C) \& (\mu(C') < \mu(C))$ *is called the probabilistic inference relation.*

Definition 19. *A rule* $C \in PR_0$, *such that* $\forall C' \in PR_0 (C' \triangleright C \Rightarrow C' \sqsubset C)$, *is called a probabilistic regularity (P-rule).*

Let PR(M) denote the set of all P-rules, and P(M) $\subset$ PR(M) – the set of all P-rules, where the premise contains at least one atom.

Definition 20. *A set of rules* $PR(M, N) = P(M) \cup D(N)$, *where* $D(N) \in D$, *and N is a certain model chosen arbitrarily from G in accordance with the measure* μ, *is called a probabilistic logic program synthesized inductively by data D(N) and the probabilistic model of data M .*

12 Predictions Based on Semantic Probabilistic Inference

Definition 21. [16]. A maximal sequence of rules $C_1 \sqsubset C_2 \sqsubset ...$, where C_1, C_2, ... $\in P(M)$, $C_i = A_i \leftarrow B^i_1, ..., B^i_{k_i}$, i = 1, 2, ... such that an atom A is unified

with all atoms A_1, A_2,... is called a *semantic probabilistic inference* (*P-inference*) *of the atom* A of the signature Ω. If such a sequence for the atom A does not exist, then the P-inference is empty. Each P-inference produces a sequence of substitutions θ_1, θ_2, ... from the definition of the relation $\rhd$. A substitution $\theta = \theta_1\theta_2...$ is called a *semantic probabilistic inference result* (calculation). The final rule of the finite P-inference is called a *resulting rule*.

Definition 22. [16]. By a *P-prediction* of some atom A of the signature Ω by the program $PR(M,N) = P(M) \cup D(N)$ we mean such a P-inference $C_1 \sqsubset C_2 \sqsubset ... \; C_i \sqsubset ... \; , C_1, C_2, ..., C_i, ... \in P(M)$ of the goal A, where:

1. There exists a rule $C_i = A_i \leftarrow B^i_1,...,B^i_{l_i}$ and a substitution θ such that $\{B^i_1\theta,...,B^i_{l_i}\theta\} \subseteq D(N)$; $A\theta = A_i\theta$; $\mu(A_i\theta) < \mu(C_i)$;
2. A maximum of conditional probability is reached on the rule C_i among all rules, satisfying condition 1, of all P-inferences of the goal A;
3. If there is no P-inference of the goal A or there is no required substitution, then a P-prediction is not defined;
4. A substitution $\theta_p = \theta_1\theta_2...\theta_{i-1}\theta$, where $\theta_1,\theta_2,...,\theta_{i-1}$ are the substitutions of the P-inference $C_1 \sqsubset C_2 \sqsubset ... \sqsubset C_i$, is called the *P-prediction result*. The value $\eta_p(A) = \mu(C_i)$ is called the *P-prediction value*. If a P-prediction is not defined, then the value $\eta_p(A)$ is not defined.

Proposition 6. *Rules satisfying point 2 are not comparable with respect to the relation* $\rhd$.

Though for the sake of simplicity the following statements (Lemmas 4–5, Corollary 11 and Theorem 7) are given in the frame of finite signature with no functional symbols (predicate and constant symbols are allowed) which is standard in probabilistic logic programming [12], a broader case of signature with functional symbols is also investigated (see [18]).

Lemma 4. *A P-prediction is defined iff there is at least one rule* $C \in P(M)$ *satisfying point 1 of Definition 22.*

Lemma 5. *Let rule* $C \in PR_0$ *satisfy point 1 of Definition 22 and has at least one atom in the premise, then either C is a P-rule* ($C \in P(M)$) *or there exists a rule* $C' \in P(M)$ *such that* $C' \rhd C$, $\mu(C') \geq \mu(C)$ *and* C' *satisfying point 1 of Definition 22.*

Corollary 11. *A P-prediction is defined iff there is a rule* $C \in PR_0$ *with at least one atom in the premise that satisfies point 1 of Definition 22.*

Let Pr be the logical program with facts belonging to the facts $D(N)$ of the program $PR(M,N) = P(M) \cup D(N)$.

Theorem 7. [16]. If atom A is predicted by the program Pr with estimation $\eta(A) > \mu(A\theta)$, for any $\theta \in \Theta G$, then it is P-predicted by the program $PR(M,N)$ with P-prediction value $\eta_p(A) \geq \eta(A)$.

13 The Relational Data Mining and Program System 'Discovery'

Based on the semantic probabilistic inference the Relational Data Mining (RDM) approach to the intensive area of applications – Knowledge Discovery in Data Bases and Data Mining (KDD&DM) – was developed in [7], [8], [13], [14], [15], [17]. The program system 'Discovery', which utilizes this approach, has been implemented. This system realizes the SP-inference and can discover the sets of laws $\mathcal{L}$, $\mathcal{LP}$ and the sets SPL, MSR. So, we may discover the full (in the sense of Theorem 1 and Propositions 2,4) and consistent (in the sense of Theorem 4) sets of rules. In [7], [17] we argue that using RDM we may cognise the object domain. The system 'Discovery' has been successfully applied to solve many practical tasks: as a cancer diagnostic systems, time series forecasting, psychophysics, bioinformatics, and many others (see www-site Scientific Discovery [19].

Acknowledgments

The work is partially supported by the Council for Grants (under RF President) and State Aid of Leading Scientific Schools (grant NSh-3606.2010.1); Russian Science Foundation grant 08-07-00272a and Integration projects of the Siberian Division of the Russian Academy of science 47,111,119.

References

1. Hempel, C.G.: Aspects of Scientific Explanation. In: Hempel, C.G. (ed.) Aspects of Scientific Explanation and other Essays in the Philosophy of Science. The Free Press, New York (1965)
2. Hempel, C.G.: Maximal Specificity and Lawlikeness in Probabilistic Explanation. Philosophy of Science 35, 116–133 (1968)
3. Salmon, W.C.: Four Decades of Scientific Explanation. University of Minnesota Press, Minneapolis (1990)
4. Tan, Y.H.: Is default logic a reinvention of inductive-statistical reasoning? Synthese 110, 357–379 (1997)
5. Krantz, D.H., Luce, R.D., Suppes, P., Tversky, A.:Foundations of measurement, vol. 1, 2, 3, p. 577 (1971) p. 493 (1986) p. 356 (1990); Acad. press, New York
6. Williamson, J.: Probability logic. In: Gabbay, D., Johnson, R., Ohlbach, H.J., Woods, J. (eds.) Handbook of the Logic of Inference and Argument: The Turn Toward the Practical. Studies in Logic and Practical Reasoning, vol. 1, pp. 397–424. Elsevier, Amsterdam
7. Vityaev, E.E., Kovalerchuk, B.Y.: Empirical Theories Discovery based on the Measurement Theory. Mind and Machine 14(4), 551–573 (2004)
8. Vityaev, E.E.: The logic of prediction. In: Proceedings of the 9th Asian Logic Conference Mathematical Logic in Asia, Novosibirsk, Russia, August 16–19, 2005, pp. 263–276. World Scientific, Singapore (2006)
9. Halpern, J.Y.: An analysis of first-order logic of probability. In: Artificial Intelligence, vol. 46, pp. 311–350 (1990)

10. Nilsson, N.J.: Probability logic. Artif. Intell. 28(1), 71–87 (1986)
11. Ng, R.T., Subrahmanian, V.S.: Probabilistic reasoning in Logic Programming. In: Proc. 5th Symposium on Methodologies for Intelligent Systems, pp. 9–16. North-Holland, Knoxville (1990)
12. Ng, R.T., Subrahmanian, V.S.: Probabilistic Logic Programming. Information and Computation 101(2), 150–201 (1993)
13. Kovalerchuk, B.Y., Vityaev, E.E.: Data Mining in finance: Advances in Relational and Hybrid Methods, p. 308. Kluwer Academic Publishers, Dordrecht (2000)
14. Kovalerchuk, B.Y., Vityaev, E.E., Ruiz, J.F.: Consistent and Complete Data and "Expert" Mining in Medicine. In: Medical Data Mining and Knowledge Discovery, pp. 238–280. Springer, Heidelberg (2001)
15. Vityaev, E.E., Kovalerchuk, B.Y.: Data Mining For Financial Applications. In: Maimon, O., Rokach, L. (eds.) Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers, pp. 1203–1224. Springer, Heidelberg (2005)
16. Vityaev, E.E.: Semantic approach to knowledge base creating. Semantic probabilistic inference of the best for prediction PROLOG-programs by a probability model of data Logic and Semantic Programming, Novosibirsk. Computational Systems 146, 19–49 (1992) (in Russian)
17. Vityaev, E.E.: Knowledge inductive inference. Computational cognition. Cognitive process modelling, p. 293. Novosibirsk State University Press, Novosibirsk (2006) (in Russian)
18. Smerdov, S.O., Vityaev, E.E.: Probability, logic & learning synthesis: formalizing prediction concept. Siberian Electronic Mathemetical Reports 9, 340–365 (2009)
19. Scientific Discovery, `http://www.math.nsc.ru/AP/ScientificDiscovery`

Visual Data Mining and Discovery in Multivariate Data Using Monotone n-D Structure

Boris Kovalerchuk[1] and Alexander Balinsky[2]

[1] Department of Computer Science, Central Washington University, 400 E. University Way, Ellensburg, WA 98926-7520, USA
[2] Cardiff School of Mathematics, Cardiff University, Senghennydd Road, Cardiff, CF24 4AG, UK

Abstract. Visual data mining (VDM) is an emerging research area of Data Mining and Visual Analytics gaining a deep visual understanding of data. A border between patterns can be recognizable visually, but its analytical form can be quite complex and difficult to discover. VDM methods have shown benefits in many areas, but these methods often fail in visualizing highly overlapped multidimensional data and data with little variability. We address this problem by combining visual techniques with the theory of monotone Boolean functions and data monotonization. The major novelty is in visual presentation of structural relations between n-dimensional objects instead of traditional attempts to visualize each attribute value of n-dimensional objects. The method relies on n-D monotone structural relations between vectors. Experiments with real data show advantages of this approach to uncover a visual border between malignant and benign classes.

Keywords: Data Mining, Visualization, Structural relations, Monotonicity, Multidimensional data, Chain.

1 Introduction

Visual data mining is an emerging research area of Data Mining and Visual Analytics [Thomas et al., 2005] to gain a deep visual understanding of data. The purpose of this paper is to develop a technique for visualizing and discovering patterns and relations from *multidimensional binary data* using the technique of monotone Boolean functions.

Visualizing the border between patterns is one of especially important aspects of visual data mining. In many situations, a user can easily catch a border visually, but its analytical form can be quite complex and difficult to discover. The simple borders of patterns that are visually far away from each other, match our intuitive concept of the pattern and increase confidence in the robustness of data mining results as discovered patterns.

Deep visual understanding is a goal of visual data mining [Beilken, Spenke, 1999]. Known visualization techniques such as parallel coordinates have been successfully used in visual data mining, but for highly overlapped multidimensional data, the

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 297–313, 2011.

resulting visualization often is unsatisfactory with a high-level occlusion. This problem is especially challenging in medical applications tracked with binary symptoms and complex dynamic optimization problems.

VDM is an especially challenging task when data richness should be preserved without excessive aggregation [Keim e al., 2002]. Another challenge is a lack of natural 3-D space and time dimensions in many tasks [Groth, 1998] that requires the visualization of abstract features.

Quite often visual representations suffer from subjectivity, poor scalability, inabilities to perceive more than 6-8 dimensions, and the slow interactive examination of complex data [Last, Kandel, 1999; Keim et al., 2002].

Glyph or *iconic visualization* is an attempt to encode multidimensional data using parameters such as the shape, color, transparency, and orientation of 2-D or 3-D objects (2-D icons, cubes, or more complex *"Lego-type"* 3-D objects) [Ebert et al, 1996; Post, van Walsum et al., 1995; Ribarsky et al., 1994]. Glyphs can visualize *nine attributes* (three positions *x*, *y*, and *z*; three size dimensions; color; opacity; and shape). Texture can add more dimensions. Shapes of the glyphs are studied in [Shaw, et al., 1999], where it was concluded that with large super-ellipses, about 22 separate shapes can be distinguished on the average.

An overview of multivariate glyphs is presented in [Ward, 2002]. Glyph methods commonly use data dimensions as positional attributes (x,y,z) to place glyphs, which often result in occlusion.

Other glyph methods place glyphs using *implicit or explicit structure* within the data set. In this paper, we show that the placement based on the use of the *data structure* is a promising approach to visualize a border between patterns for multidimensional data. We call this the **GPDS** approach (*Glyph Placement on a Data Structure*). In this approach, some attributes are *implicitly* encoded in the data structure while others are *explicitly* encoded in the glyph/icon. Thus, if the structure carries ten attributes and a glyph/icon carries nine attributes, nineteen attributes are encoded.

We use simple 2-D icons as *bars* of different colors. Adding texture, motion and other icon characteristics can increase dimensions of data visualized. Collapsing of the bar to a single pixel is another way to make GPDS approach more scalable**.**

Alternative techniques such as g*eneralized spiral and pixel bar* chart are developed in [Keim et al., 2002]. These techniques work with large data sets without overlapping, but only with a few attributes (≤6). Other visualization methods, known as Scatter, Splat, Map, Tree, and Evidence Visualizer, that are implemented in MineSet (Silicon Graphics), permit up to eight dimensions to be shown on the same plot by using color, size, and animation of different objects [Last, Kandel, 1999].

The *parallel coordinate technique* [Inselberg, Dimsdale, 1990] can work with ten or more attributes, but suffers from record overlap and thus is limited to tasks with well-distinguished cluster records. In parallel coordinates, each vertical axis corresponds to a data attribute (x_i) and a line connecting points on each parallel coordinate corresponds to a record.

Parallel coordinates visualize explicitly *every* attribute x_i of an n-dimensional vector $(x_1,x_2,\ldots,x_n)$ in 2-D and place the vector using *all attributes* x_i, but each attribute is placed on its own parallel coordinate *independently* of placing other

attributes of this vector and other vectors. This is one of the major reasons of occlusion and overlap of visualized data. The GPDS approach constructs a data structure and can place objects using *attribute relations*.

A typical example of current research in high-dimensional Visual Analytics is the work of Wilkinson et al. [2006], which uses the 2-D *distributions of pairwise orthogonal projections* of the multidimensional Euclidean space. 2-D projections may not discover real *n*-D patterns and glyphs often lead to occlusion.

We attempt to show only relevant structural relations between n-D vectors in 2-D or 3-D, not actual n-D vectors. A more extensive representation of related work is presented in [Peng et al., 2004; Mackinlay, 1997; de Oliveira, Levkowitz, 2003; Yang et al., 2007].

The proposed method to represent n-D data in 2-D or 3-D is based on the following principles of visual representations, learning, and discovery.

Principle 1: *Represent complexity, not abstract multiplicity*. The visual system was developed evolutionary to deal efficiently with dynamics of complex concrete objects in 3-D world. This ability does not imply the same efficiency to deal with multiple abstract multidimensional objects.

Principle 2: *Represent concrete complexity in low dimensions*. The human visual system has unique abilities to understand concrete complex information received via visual channels in low dimensions (2-D and 3-D). Therefore, it is desirable to transform the n-D data to concrete 2-D or 3-D entities.

Principle 3: *Represent individual attributes of n-D data in 2-D/3-D only if necessary*. This will help to minimize occlusion and loss of information. The parallel coordinates method that represents all attributes of n-D data suffers from occlusion.

Principle 4: *Represent structural relations between n-D data in 2-D or 3-D that have a clear meaning for the analyst*. Such an attempt can avoid occlusion and loss of information.

Principle 5: A human can *capture structural relations such as order (hierarchy, generalization) and monotonicity in n-D*. Therefore, these relations are first candidates to be a base for n-D data representation in 2-D and 3-D.

Principle 6: *If n-D data have no recorded natural hierarchy and n-D monotonicity, modify data representation, learn, discover, and build these structures* in a new representation with clear meaning for the analyst.

This paper is organized as follows. Section 2 contains definitions and mathematical statements. Section 3 describes the proposed visual representation of the n-D Boolean space in 2-D. Section 4 describes a learning process and section 5 presents results of a computational experiment.

The paper concludes with summary of the results and an outline of the further research.

2 Chains and Similarity Distances between Chains

We denote the set of all *n*-D binary vectors $E^n := \{0,1\}^n$ (*n*-D binary cube), which is a partially ordered set that forms a lattice with the greatest element (1,1,…,1) and the smallest element (0,0,…,0) for the relation $\leq$ on vectors $\mathbf{a}=(a_{1,\ldots,}a_n) \in E^n$ and $\mathbf{b}=(b_{1,\ldots,}b_n) \in E^n$, I={1,2,…,n},

$$\mathbf{a} \leq \mathbf{b} \Leftrightarrow \forall\, i \in I\ a_i \leq b_i$$

Only some vectors in E^n are ordered in this way.

Consider a pair (M, ≤), where M is a set of elements and "≤" is a partial order relation on M.

Definition. A subset C ⊆ M is called a *chain* if any two elements **g**, **h** of C are comparable, i.e., satisfy linearity property, **g** ≤ **h** or **h** ≤ **g** [Birkhoff,1995; Grätzer, 1971].

In our case $M=E^n$ with the partial order defined above.

Definition. Let **a**, **b**, **c**, and **d** be *n*-D Boolean vectors then a triple <**a**,{**b**,**c**},**d**> is called a *square* (see Fig. 1a) if **a<b<d**, **a<c<d, b≠c,** and |**a**|+1=|**b**|=|**c**|=|**d**|-1, where | . | is the *Hamming norm*, $|\mathbf{x}| = \sum_{i=1}^{n} x_i$, that is the number of 1s in the vector.

Vectors **b** and **c** are incomparable relative to ≥ which is denoted as **b**||**c** [Grätzer, 1971]. The concept of square for E^n was introduced in [Hansel, 1966] in slightly different terms. Below we introduce the concepts of adjacent and S-adjacent squares.

Definition. Squares S_1 = <$\mathbf{a}_1$,{$\mathbf{b}_1$,$\mathbf{c}_1$},$\mathbf{d}_1$> and S_2 = <$\mathbf{a}_2$,{$\mathbf{b}_2$,$\mathbf{c}_2$},$\mathbf{d}_2$> are called *adjacent* iff |{b_1,c_1} ∩ {b_2,c_2}| = 1. See Fig. 1b where $\mathbf{c}_1 = \mathbf{b}_2$.

Definition. Chains C_1 and C_2 are *S-adjacent where S:*= <**a**,{**b**,**c**},**d**> is a square, if one of them contains three elements **a**, **b** and **d** of the square S and the other one contains the forth element **c** of S.

C_1 and C_2 are called *adjacent* if there is a square S such that they are S-adjacent.

In Table 1 chains 00H and 01H are adjacent, because these chains are S_1-adjacent, that is they contain square S_1 = <$\mathbf{a}_1$,{$\mathbf{b}_1$,$\mathbf{c}_1$},$\mathbf{d}_1$> = <(000),{(001), (010)},(011)>. Similarly, chains 00H and 10H are also adjacent, because they contain another square S_2 = < $\mathbf{a}_2$,{$\mathbf{b}_2$,$\mathbf{c}_2$},$\mathbf{d}_2$> = <(001), {(011), (101)}, (111) >.

In the same way in Table 2, chains 000H and 100H are adjacent with S=<(0011),{(0111), (1011)},(1111)>, and in Table 3 chains 0000H and 1000H are adjacent with S=<(00111),{(01111), (10111)},(11111)>.

Definition. A set of chains {Q} in E^n is called a *partitioning set of chains (or P-set of chains)* if {Q} *partitions* E^n, i.e., chains in {Q} do not overlap and cover E^n completely.

The partitioning property is important for visual data mining of multidimensional data because it helps to avoid occlusion of n-D Boolean data, visualized in 2-D or 3-D as we show below in our design of the visual representation in section 3.

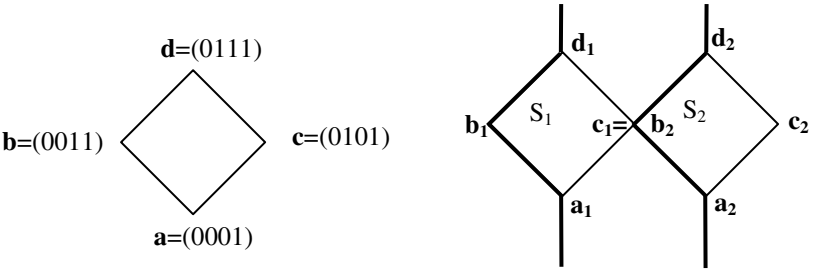


Fig. 1. Examples of a Boolean square (a) and adjacent squares (b)

Fig. 1 illustrates a square and a pair of adjacent squares. Below we define Hansel chains [Hansel, 1966, Kovalerchuk et al., 1996] in E^n, which is a special type of P-set of chains with multiple useful properties. These properties of Hansel chains have been used to identify the minimal number of tests/questions needed to restore any monotone Boolean function [Hansel, 1966].

Hansel chains for E^n are defined recursively from chains in E^{n-1}. Therefore, we start from defining Hansel chains in E^1.

Definition. The single Hansel chain in $E^1=\{0,1\}$ is defined as H:={0,1}, hence the set of Hansel chains in E^1 is the partition with a single class {H}.

These vectors are ordered by their Hamming norms, |1|> |0|.

Definition. The set of Hansel chains in E^2 consists of the two chains 1H:={10} and 0H:={00,01,11}. These chains partition E^2. We will extend this notation later.

Note that the set of chains U={{01},{00,10,11}} is also a partitioning set of chains, but U is not the set of Hansel chains in E^2.

Definition. The set of Hansel chains {C} in E^3 consists of chains presented in Table 1. These chains also do not overlap and cover E^3 completely.

Table 1. The Hansel chains in E^3

	Chain	*Norm 0*	*Norm 1*	*Norm 2*	*Norm 3*
1	00H	000	001	011	111
2	10H		100	101	
3	01H		010	110	

The *general recursive process* of defining *a set of Hansel chains* in E^{n+1} from a set of Hansel chains in E^n is as follows [Hansel, 1966; Kovalerchuk et al., 1996].

The main steps of the recursive process are:

(1) *producing* two intermediate chains in E^{n+1} from each Hansel chain in E^n, and
(2) *modifying these intermediate chains* to produce Hansel chains in E^{n+1}.

To get the first intermediate chain, each element of a chain from E^n is expanded by adding a leading zero, and to get the second intermediate chain each element of the same chain from E^n is expanded by adding a leading one.

In particular, having a single chain H={0,1} in E^1 we get intermediate chains in E^2, 0H'={00, 01} and 1H'={10, 11} with two Boolean vectors each. Then the growth-cut process is applied to produce final Hansel chains in E^2: 0H={00,01,11} and 1H={10} with three elements and one element, respectively. The growth-cut process cuts the largest element of 1H' that is (11) and grows 0H' by adding (11) as a new top element to 0H' to get 0H={00,01,11}.

For building Hansel chains in E^3 from 0H={00,01,11} in E^2 we get first 00H'= {000,001,011}. Next, we produce a chain 10H' with leading 1 from H, 10H'= {100,101,111}.

The *growth-cut step* cuts the greatest element from 10H' and grows 00H' by adding the greatest element to 00H. This produces chains 00H={000,001,011,111} and 10H={100,101} with 4 and 2 elements, respectively.

We have shown how chain 0H={00,01,11} produces chains in E^3. Similarly, chain 1H={10} that consists of a single element produces first two preliminary chains in E^3: 01H'={010} and 11H'={110} that also consist of single elements. Then the growth-cut process produces the final chains 01H={010,110} and 11H={} which is empty and deleted.

The application of these steps in E^3 produces the set of Hansel chains in E^4 (see Table 2), and from these we get a set of Hansel chains in E^5 (see Table 3). All Hansel chains of the recursively constructed sets of Hansel chains for higher n are produced in the same way.

Table 2. The set of Hansel chains in E^4

	Chain	*Norm 0*	*Norm 1*	*Norm 2*	*Norm 3*	*Norm 4*
1	000H	0000	0001	0011	0111	1111
2	100H		1000	1001	1011	
3	010H		0100	0101	1101	
4	110H			1100		
5	001H		0010	0110	1110	
6	101H			1010		

Table 3. The set of Hansel chains in E^5

	Chain	*Norm 0*	*Norm 1*	*Norm 2*	*Norm 3*	*Norm 4*	*Norm 5*
1	0000H	00000	00001	00011	00111	01111	11111
2	1000H		10000	10001	10011	10111	
3	0100H		01000	01001	01011	11011	
4	1100H			11000	11001		
5	0010H		00100	00101	01101	11101	
6	1010H			10100	10101		
7	0110H			01100	11100		
8	0001H		00010	00110	01110	11110	
9	1001H			10010	10110		
10	0101H			01010	11010		

Empty chains 1110H and 11010 are omitted in Table 3.

Statement 1. In any Hansel chain C in E^n constructed above recursively with elements $\mathbf{h}_i$, C={ $\mathbf{h}_i$ | i=1:k}, the labelling of the elements can be chosen in such a way that the *one-step property* $|\mathbf{h}_{i+1}|=|\mathbf{h}_i|+1$ is satisfied.

Proof. We prove by induction that for each integer n≥1 the one-step-property of each Hansel chain in the set of Hansel chains in E^n holds. It is obvious that this holds for n=1. Let n≥1. We now assume the induction hypothesis that the one-step-property of each Hansel chain in the set of Hansel chains in E^n holds and prove this statement for n+1. Let C:={$\mathbf{h}_1,\ldots, \mathbf{h}_k$}, k≥1, be a Hansel chain in E^{n+1}. We use from the recursive construction that C is either of the form (0) C={$0\mathbf{f}_1,\ldots, 0\mathbf{f}_{k-1}, 1\mathbf{f}_{k-1}$} where {$\mathbf{f}_1,\ldots, \mathbf{f}_{k-1}$} is a Hansel chain in E^n or (1) C={$1\mathbf{g}_1,\ldots,1\mathbf{g}_{k-1}, 1\mathbf{g}_k$} where {$\mathbf{g}_1,\ldots, \mathbf{g}_{k+1}$} is a Hansel chain in E^n. In either case we get from the one-step-property of the Hansel chains in E^n that C has the one-step-property.

Statement 2. If chains C_1 and C_2 are S_1-adjacent in such a way that square $S_1=<\mathbf{a}_1,\{\mathbf{b}_1,\mathbf{c}_1\}, \mathbf{d}_1>$ contains elements $\mathbf{a}_1$, $\mathbf{b}_1$, $\mathbf{d}_1$ in chain C_1 and element c_1 in chain C_2 and square $S_2=<\mathbf{a}_2,\{\mathbf{b}_2,\mathbf{c}_2\},\mathbf{d}_2>$ contains elements $\mathbf{a}_2$, $\mathbf{b}_2$, $\mathbf{d}_2$ in C_2 and $\mathbf{c}_1=\mathbf{b}_2$ (see Fig. 1b), then $|\mathbf{a}_1-\mathbf{a}_2|=|\mathbf{b}_1-\mathbf{c}_1|=|\mathbf{b}_1-\mathbf{b}_2|=|\mathbf{d}_1-\mathbf{d}_2|=2$.

Proof. According to the definition of square S_i, $|\mathbf{a}_i|=|\mathbf{c}_i|-1=|\mathbf{b}_i|-1$, i=1,2, therefore $|\mathbf{a}_i-\mathbf{c}_i|=|\mathbf{a}_i-\mathbf{b}_i|=1$. Also $\mathbf{c}_1=\mathbf{b}_2$, thus $|\mathbf{c}_1|=|\mathbf{b}_2|$ and $|\mathbf{a}_1|=|\mathbf{a}_2|$. Thus, $|\mathbf{a}_1|=|\mathbf{a}_2|=|\mathbf{c}_1|-1$. Also $\mathbf{a}_1\neq\mathbf{a}_2$ because $\mathbf{a}_1$ and $\mathbf{a}_2$ belong to different chains. Vector $\mathbf{a}_1$ alters $\mathbf{c}_1$ in one attribute and vector $\mathbf{a}_2$ alters $\mathbf{c}_1$ it in another attribute. Say, $c_{1i}=1$ and $c_{1j}=1$, then $a_{1i}=0$, and $a_{2j}=0$. Thus, the distance between $\mathbf{a}_1$ and $\mathbf{a}_2$ is 2, $|\mathbf{a}_1-\mathbf{a}_2|=2$. Similarly, $\mathbf{d}_1$ and $\mathbf{d}_2$ alter $\mathbf{c}_1$ in two positions, thus, $|\mathbf{d}_1-\mathbf{d}_2|=2$.

Statement 3. For all $\mathbf{x}$, $\mathbf{y}$ such that $\mathbf{x}\neq\mathbf{y}$ and $|\mathbf{x}|=|\mathbf{y}|$ the distance $|\mathbf{x}-\mathbf{y}|\geq 2$ and $\mathbf{x}$ and $\mathbf{y}$ exist such that $|\mathbf{x}-\mathbf{y}|=2$.

This means that if two Boolean vectors differ but have equal norms, then the minimal Hamming distance between these vectors is equal to 2.

Proof. If this distance would be equal to 1, $|\mathbf{x}-\mathbf{y}|=1$, then it will be only one *i* such that $x_i\neq y_i$, say $x_i=0$ and $y_i=1$. This will imply that $|\mathbf{x}|<|\mathbf{y}|$ which contradicts the condition that $|\mathbf{x}|=|\mathbf{y}|$.

Statements 2 and 3 help us to clarify and define the distance between Hansel chains using the distance between their elements. Say, we can measure the distance between two Hansel chains as a Hamming distance between their closest elements. Statement 2 tells us that there are elements of adjacent chains with the distance equal to 2 and statement 3 tells us that the distance between elements of different chains cannot be less than 2. Thus, the adjacent Hansel chains C_1 and C_2 are closest Hansel chains in this measure of the distance between Hansel chains.

Theorem. If C is a Hansel chain in E^n ($n \geq 1$) with $k \geq 2$ elements, then 0C and 1C (as constructed above) are adjacent Hansel chains in E^{n+1} and

$$\forall\, \mathbf{h}_1 \in 0C\ \forall\, \mathbf{h}_2 \in 1C\ (|\mathbf{h}_1|=|\mathbf{h}_2| \Rightarrow |\mathbf{h}_1-\mathbf{h}_2|=2). \quad (1)$$

In other words, the distance between vectors of the same norm in Hansel chains 0C and 1C is the constant 2 and this is the smallest distance for different Hansel chains. This is a reason to locate 0C and 1C next to each other in the visualization process.

Proof. Let $C=\{\mathbf{g}_1,\mathbf{g}_2,\ldots,\mathbf{g}_k\}$ be a Hansel chain in E^n ($n \geq 1$) and $k \geq 2$. By the one-step-property we assume that $|\mathbf{g}_i - \mathbf{g}_{i-1}|=1$ for $2 \leq i \leq k$.

Then $0C=\{0\mathbf{g}_1,0\mathbf{g}_2,\ldots,0\mathbf{g}_{k-1},0\mathbf{g}_k,1\mathbf{g}_k\}$ and $1C=\{1\mathbf{g}_1,1\mathbf{g}_2,\ldots,1\mathbf{g}_{k-1}\}$, and for the square $S:=<0\mathbf{g}_{k-1},\{0\mathbf{g}_k,1\mathbf{g}_{k-1}\},1\mathbf{g}_k>$ the Hansel chains 0C and 1C are S-adjacent. This is illustrated below with the square S shown in grey.

chain	*norm* $\lvert\mathbf{g}_1\rvert$	*norm* $\lvert\mathbf{g}_1\rvert+1$	*norm* $\lvert\mathbf{g}_1\rvert+2$		*norm* $\lvert\mathbf{g}_1\rvert+i-1$		*norm* $\lvert\mathbf{g}_1\rvert+k-2$	*norm* $\lvert\mathbf{g}_1\rvert+k-1$	*norm* $\lvert\mathbf{g}_1\rvert+k$
0C	$0\mathbf{g}_1$	$0\mathbf{g}_2$	$0\mathbf{g}_3$	…	$0\mathbf{g}_i$	…	$0\mathbf{g}_{k-1}$	$0\mathbf{g}_k$	$1\mathbf{g}_k$
1C		$1\mathbf{g}_1$	$1\mathbf{g}_2$	…	$1\mathbf{g}_{i-1}$	…	$1\mathbf{g}_{k-2}$	$1\mathbf{g}_{k-1}$	
C	$\mathbf{g}_1$	$\mathbf{g}_2$	$\mathbf{g}_3$		$\mathbf{g}_i$		$\mathbf{g}_{k-1}$	$\mathbf{g}_k$	

Let $\mathbf{h}_1 \in 0C$ and $\mathbf{h}_2 \in 1C$ and $|\mathbf{h}_1|=|\mathbf{h}_2|$, then there exists i such that $2 \leq i \leq k$ and $\mathbf{h}_1=0\mathbf{g}_i$ and $\mathbf{h}_2=1\mathbf{g}_{i-1}$, hence $|\mathbf{h}_1-\mathbf{h}_2| = |0\mathbf{g}_i - 1\mathbf{g}_{i-1}|=1+1=2$, since $|\mathbf{g}_i - \mathbf{g}_{i-1}|=1$ by the one-step-property of C.

This theorem is illustrated in Tables 4-6 with actual distances computed for E^3-E^5.

Statement 4. The distance between *n*-D Boolean vectors $\mathbf{a}=\{a_j\}$ and $\mathbf{b}=\{b_j\}$ with equal norms, $|\mathbf{a}|=|\mathbf{b}|$, is an even number.

Proof. If r is the number of bits j where $a_j=1$ and $b_j=0$, then there should be r other bits where $a_j=0$ and $b_j=1$ to have $|\mathbf{a}|=|\mathbf{b}|$. In this case the total number of bits where **a** and **b** differ is 2r, which is their Hamming distance, $|\mathbf{a}-\mathbf{b}|=2r$.

Statement 5. If C_1 and C_2 are S_1-adjacent Hansel chains, and C_2 and C_3 are S_2-adjacent Hansel chains such that

$$S_1=<\mathbf{a},\{\mathbf{b},\mathbf{c}\},\mathbf{d}>,\ \mathbf{a},\mathbf{b},\mathbf{d}\in C_1,\quad S_2=<\mathbf{e},\{\mathbf{c},\mathbf{f}\},\mathbf{g}>,\ \mathbf{c},\mathbf{e},\mathbf{g}\in C_2,\ \mathbf{f}\in C_3$$

then $|\mathbf{b}|=|\mathbf{f}|$ and the Hamming distance $|\mathbf{b}-\mathbf{f}|$ is equal to 2, or 4.

Proof. The properties $|\mathbf{b}|=|\mathbf{c}|$, $|\mathbf{c}|=|\mathbf{f}|$, $|\mathbf{b}-\mathbf{c}|=2$, and $|\mathbf{c}-\mathbf{f}|=2$ of squares S_1 and S_2 are derived from the definition of the square. Thus, $|\mathbf{b}|=|\mathbf{f}|$. Next, the Hamming distance as every distance satisfies a triangle inequality: $|\mathbf{b}-\mathbf{f}| \leq |\mathbf{b}-\mathbf{c}|+|\mathbf{c}-\mathbf{f}|=2+2=4$. The distance $|\mathbf{b}-\mathbf{f}|$ cannot be the odd number 3 (see Statement 3).

Now we investigate the following conjecture:

Constant Distance Conjecture: For all Hansel chains C_1, C_2 in E^n there exists a constant c, such that $|\mathbf{h}_1-\mathbf{h}_2|=c$ for all $\mathbf{h}_1 \in C_1$ and $\mathbf{h}_2 \in C_2$ with equal norms, $|\mathbf{h}_1|=|\mathbf{h}_2|$.

We explore for which *n* this Conjecture is true.
It is true for *n*=1, 2, 3, and 4. See Tables 4 and 5 for *n*=3 and *n*=4. This means that we can build a very consistent visual representation of Hansel chains for these *n*.

Table 6 shows that is not true for n=5, that provides counterexamples for n=5. In Table 4 notation 22 means two distances, e.g., for chains 00H and 10H, the first 2 is the distance between (001) and (100), and the second 2 is the distance between (011) and (101) from chains 00H and 10H, respectively.

In Table 6, notation 2242 indicates 4 distances for 4 elements of chains 2 and 5, where the distances between vectors (10011) and (01101) is equal to 4 and the distances between vectors (10001) and (00101) is equal to 2. This provides a counterexample.

Table 4. Distances between Hansel chain elements with equal norms in E^3

	Chain	*1 Chain 00H*	*2 Chain 10H*	*3 Chain 01H*
1	Chain 00H	0	22	22
2	Chain 10H		0	22
3	Chain 01H			0

Definition. Let $D_{HC}(H_1, H_2) = (\sum_{|\mathbf{h}_1|=|\mathbf{h}_2|} |\mathbf{h}_1 - \mathbf{h}_2|) / m$,

where $\mathbf{h}_1$ and $\mathbf{h}_2$ are elements of chains H_1 and H_2, respectively, with the same Hamming norm $|\mathbf{h}_1|=|\mathbf{h}_2|$, and m is the number of such pairs of elements. D_{HC} is called the *averaged Hamming chain similarity measure*.

Table 7 presents this measure for E^5.

Table 5. Distances between all Hansel chain elements with the same norm in E^4

Chain	*1*	*2*	*3*	*4*	*5*	*6*
1	0	222	222	4	222	2
2		0	2	2	4	2
3			0	2	2	2
4				0	2	2
5					0	2
6						0

Table 6. Distances between all Hansel chain elements with the same norm in E^5

Chain	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	0	2222	2222	44	2222	42	44	2222	22	24
2		0	2222	22	2242	42	44	2422	22	24
3			0	22	2222	42	22	2422	24	22
4				0	44	22	22	44	24	22
5					0	22	22	22	44	44
6						0	22	24	22	22
7							0	22	22	22
8								0	22	22
9									0	22
10										0

Table 7. Averaged Hamming chain similarity measure D_{HC} between Hansel chains in E^5

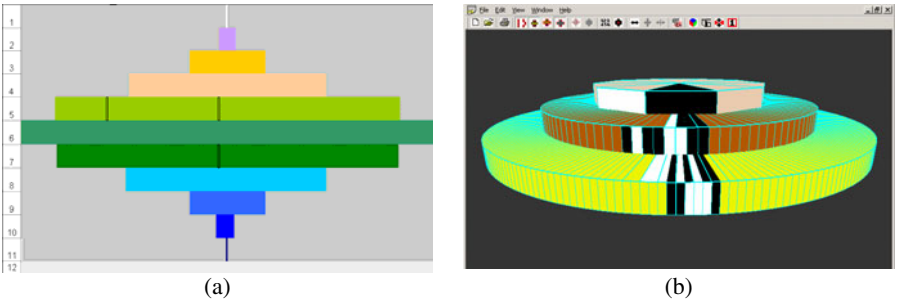
Chain	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	0	2	2	4	2	3	4	2	2	3
2		0	2	2	2.5	3	4	2.5	2	3
3			0	2	2	3	2	2.5	3	2
4				0	4	3	2	4	3	2
5					0	2	2	2	4	4
6						0	2	3	2	2
7							0	2	2	2
8								0	2	2
9									0	2
10										0

3 Representing and Drawing of n-D Boolean Space in 2-D

Below we describe a structure to allocate Boolean vectors in 2-D. Boolean vectors are represented in the 2-D plane, such that their position in the plane grows with its Boolean norm.

Each vector will be first placed in accordance with its norm (level) and then drawn as a colored bar: white for the 0 class, black for the 1 class. Fig. 2 depicts the hierarchy of levels of Boolean vectors in E^{10}, which is 11 levels from 0 to 10, where level 0 contains a single vector (0000000000) and level 10 contains a single vector (1111111111). Several vectors are shown in Fig. 2 as *small bars*.

The vectors on the third row cannot be identified from this figure without additional assumptions, but we can tell for sure that it has eight "1"s, because of its location on the third line that contains all vectors with norm 8. To specify it more we can link each vertical position (column) with a specific vector. Assume that Boolean vectors with the same norm are ordered as decimal numbers, where the leftmost column presents the largest decimal vector for the given norm, (1111111100), and the rightmost position is used for vector (0011111111) in the third row. Each level in Fig. 2 is called a disk and the entire visualization is called the **multiple disk form** (**MDF**). In 2-D disks are just rectangles, but in 3-D each level is an actual disk (see Fig. 2b).



(a) (b)

Fig. 2. (a) 2-D Multiple Disk Form (MDF) representation of 10-dimensional Boolean space without occlusion. (b) A 3-D version of MDF with grouping Hansel chains.

There is *no occlusion* of vectors in MDF. This is the fundamental advantage of the MDF representation in comparison with methods reviewed in section 1. In Fig. 2, all 1024 10-D Boolean vectors are completely visible and their attributes are represented implicitly but unambiguously.

The advantage of this procedure is to allow the user to compare several binary datasets or Boolean functions at a time side-by-side. This representation uses a distance between vectors as decimal numbers, $D_N(n(\mathbf{a}),n(\mathbf{b})) = |n(\mathbf{a})-n(\mathbf{b})|$, where $n(\mathbf{a})$ and $n(\mathbf{b})$ are decimal numbers representing vectors a and b.

The disadvantage of such MDF implementation, called *process* P_0, is that distance D_N may not be relevant to the application domain. Thus, the task is to locate vectors in MDF in a way that will capture relations or structure that are *relevant* to the domain. The *Hamming distance* D_H captures relevant relations in many domains. A partial order relation $\leq$ between vectors $\mathbf{a} \leq \mathbf{b} \Leftrightarrow \forall a_i \leq b_i$ is one that represents some ordinal structure of the attribute space. The following statement shows the link between D_H and this partial order.

Statement 6. If $D_H(\mathbf{a},\mathbf{b})=1$, then $\mathbf{a}<\mathbf{b}$ or $\mathbf{b}<\mathbf{a}$. If $\mathbf{a}<\mathbf{b}$ and $D_H(\mathbf{a},\mathbf{b})=k$, then a chain exist such that there are k-1 elements between **a** and **b** on this chain.

Below we describe a process of allocation and relocation of vectors in MDF based on Hansel chains that differs from process P_0 described above.

Algorithm

This new algorithm uses the statements and the theorem from section 2 and has the following major steps:

(1) Putting the largest chain C into the center of the MDF,

(2) Ordering other chains relative to their closeness to C in the *averaged Hamming chain similarity measure,* $D_{HC}(U, C)$ defined in section 2,

(3) Ordering chains with equal *similarity measure* D_{HC} to C relative to the location of the borders between patterns on these chains.

(4) Locating chains in MDF in accordance with their order obtained in (2) and (3), that is the closest chains are located next to the largest chain first and all other chains are located next according their order. If two chains have the same distance D_{HC} to the largest chain then the chain that has a border closer to the largest chain is located first.

Tables 8-11 show chains for E^1-E^5 located using this algorithm without step (3).

Table 8. Horizontal drawing of Hansel chains in E^1

Chain	*Norm 0*	*Norm 1*
H	0	1

Table 9. Horizontal drawing of Hansel chains in E^2

Chain	*Norm 0*	*Norm 1*	*Norm 2*
0H	00	01	11
1H		10	

Table 10. Horizontal drawing of Hansel chains in E^3

Chain	*Norm 0*	*Norm 1*	*Norm 2*	*Norm 3*
01H		010	110	
00H	000	001	011	111
10H		100	101	

Table 11. Horizontal drawing of Hansel chains in E^4 in accordance with chain distances

Chain	*Norm 0*	*Norm 1*	*Norm 2*	*Norm 3*	*Norm 4*
101H			1010		
010H		0100	0101	1101	
000H	0000	0001	0011	0111	1111
100H		1000	1001	1011	
001H		0010	0110	1110	
110H			1100		

Table 12. Horizontal drawing of Hansel chains in E^5 in accordance with chain distances

Chain C	*Distance $D_{HC}(C,0000H)$*	*Norm 0*	*Norm 1*	*Norm 2*	*Norm 3*	*Norm 4*	*Norm 5*
0110H	4			01100	11100		
0101H	3			01010	11010		
1001H	2			10010	10110		
0001H	2		00010	00110	01110	11110	
0100H	2		01000	01001	01011	11011	
0000H	0	00000	00001	00011	00111	01111	11111
1000H	2		10000	10001	10011	10111	
0010H	2		00100	00101	01101	11101	
1010H	3			10100	10101		
1100H	4			11000	11001		

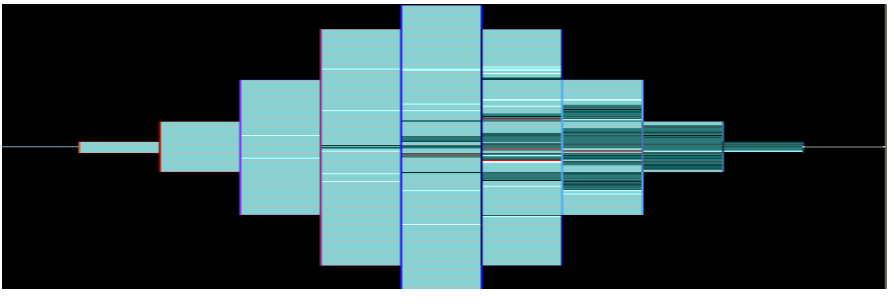


Fig. 3. MDF in E^{10} with Hansel chains applied to cancer data: white bars represent benign tumor cases, black bars represent cancer tumor cases

Tables 9-12 can be converted to the visual form as shown in Fig. 3 for E^{10}. It is the MDF from Fig. 2a rotated with chains of 10-D vectors from a cancer diagnostics application. In Fig. 3, the white bars show benign tumor cases, the black bars cancer tumor cases.

In the previous consideration [Kovalerchuk, Delizi, 2005] any relocation of chains in MDF was allowed. This means that Boolean vectors from different chains that are not similar can be located next to each other. That method does not control and prevents such occurrences. The vicinity of some Boolean vector **a** may contain both vectors that are similar to **a** (being on the same chain) and be dissimilar being on the other chains. In other words, it captures relations (similarity and monotonicity) between elements along the chains, but does not ensure similarity for Boolean vectors from different chains.

The new algorithm imposed limitations to ensure that chains that are located next to each other are close in the averaged Hamming chain similarity measure D_{HC}. We call this requirement the *Local Similarity Principle* (LSP).

4 Learning Process by Monotone Extension

Learning algorithms in data mining and machine learning include: (1) discovering a regularity R using training and testing data for specific classes, (2) generalizing R to new cases, and (3) applying R to classify these new cases. A complete generalization will classify all cases, that is all vectors in an n-D space.

Often learning algorithms do not separate (1)-(3), but only produce a border between classes as a hyperplane or another discriminating surface in an n-D space as a generalization of R. The lack of explicitly formulated regularity R is a flaw of such algorithms because it can lead to overgeneralization. In this section, we describe a process of learning monotone regularities R (in an *n*-D Boolean space) that are explicitly defined.

Definition. A Boolean function f: $E^n \rightarrow E=\{0,1\}$ is called a *monotone Boolean function* if

$$\forall\ \mathbf{x} \geq \mathbf{y} \Rightarrow f(\mathbf{x}) \geq f(\mathbf{y}) \tag{2}$$

We call a monotone Boolean function also a *monotone regularity*, if the Boolean function is used to describe some regularity in given data.

Discovering monotone regularity

At first, we *discover monotonicity* on training data T_r by testing property (2) on all pairs of training data (vectors **x**, **y** with known R values.) The computational process can be shortened by using Hansel chains.

If monotonicity of R is confirmed on T_r, then it is tested with vectors of a set T_t disjoint from T_r. If this test is also successful, then we *generalize* R. This means that we consider R as a monotone Boolean function for all elements of E^n. Then we apply the generalized R as a monotone Boolean function to new cases by using **monotone extension** that holds for monotone functions

$$(\ \mathbf{x} \geq \mathbf{a}\ \ \&\ R(\mathbf{a})=1)\ \Rightarrow\ R(\mathbf{x})=1;\quad (\ \mathbf{a} \geq \mathbf{y}\ \ \&\ R(\mathbf{a})=0\ ) \Rightarrow R(\mathbf{y})=0; \tag{3}$$

here **a** is a vector from training data with known class R(**a**), e.g., benign or malignant, and **x** is a new vector with unknown R(**x**).

We do this extension using Hansel chains. If **a** belongs to a chain C and R(a)=1, then for all **x>a** on this chain we assign R(**x**)=1. Similarly, if **a>x** and R(**a**)=0, then we assign R(**x**)=0 for all **x** below **a** on that chain.

Classes and borders between classes

Let R: $E^n \to E$ be a monotone Boolean function.
Definition. *Class 1* is the set of $\mathbf{x} \in E^n$ such that R(**x**)=1 and *Class 0* is the set of $\mathbf{x} \in E^n$ such that R(**x**)=0.

Definition. Let C be a chain in E^n, X be a subset of C. If there exists an element $x \in X$ with R(x)=1, then the minimal element in $X \cap$ *Class 1* is denoted by $x_{min1,C}$ and is called the *lower one* in X for R. If there exists an element $x \in X$ with R(x)=0, then the maximal element in $X \cap$ *Class 0* is denoted by $x_{max0,C}$ and is called the *upper zero* in X for R.

If X=C, then $\mathbf{x}_{max0,C}$ is the upper zero and $\mathbf{x}_{min1,C}$ is the lower one of the chain C for R. In the case of X=C, $\mathbf{x}_{max0,C} < \mathbf{x}_{min1,C}$ since R is monotone. If C has the one-step-property, then $|\mathbf{x}_{max0,C} - \mathbf{x}_{min1,C}|=1$, that is $\mathbf{x}_{min1,C}$ is the next element on the chain C after $\mathbf{x}_{max0,C}$. These two elements form the *R-border between Class 0 and Class 1* on chain C, denoted by $(\mathbf{x}_{max0,C}, \mathbf{x}_{min1,C})$.

Let X be a subset of training data T_r on the chain C, $X \subseteq T_r \cap C$, such that it contains **x** and **y** elements, R(**x**)=1 and R(**y**)=0.

Definition. A Hansel chain C is called a *0-1-Hansel chain*, if an element **x** with R(**x**)=0 and an element **y** with R(**y**)=1 exist on C.

Definition. The R-border on a set of *0-1-Hansel chains* $\{C\}_{0\text{-}1}$ in E^n is the set of the R-borders of all 0-1 Hansel chains C.

Definition. The *width of the border* $(\mathbf{x}_{max0,C}, \mathbf{x}_{min1,C})$ on chain C for a subset X of C is the Hamming distance

$$|\mathbf{x}_{max0,C} - \mathbf{x}_{min1,C}|.$$

The width is always = 1, if the monotone Boolean function R is fully defined, and the chain C has the one-step-property, and X=C.

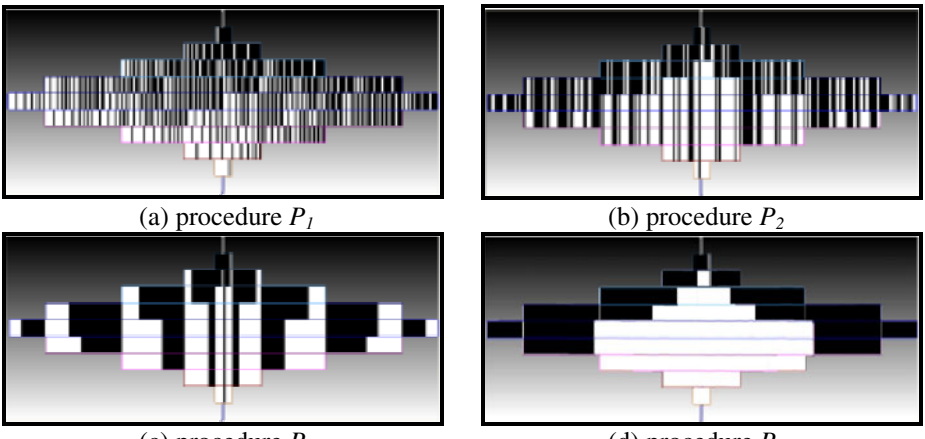
In practice often a Boolean function R is known only partially, that is only for elements of the training data T_r, $T_r \subset E^n$. Thus it is possible that $|\mathbf{x}_{max0,C} - \mathbf{x}_{min1,C}| > 1$ for some chains. A wider border between classes indicates a better separation of classes in training data T_r.

The monotone extension of R beyond the training data using (3) may decrease the width of the border and may produce a fully defined Boolean function in E^n.

5 Computational Experiment

We conducted several computational experiments that show a wide variety of borders between classes. The width and clarity of this border depends on the data and visual data mining procedures used.

Different ways of locating Boolean vectors on MDF form produce very different results relative to the shape of the border between classes (see Fig. 4). They range from no border visible (Fig. 4a) to a very clear border (Fig. 4d). Experiments with real breast cancer data show advantages of the proposed approach to uncover a visual border between benign and malignant cases in breast cancer. To be able to see a large number of attributes a user can use zooming or scrolling MDF.



(a) procedure P_1 (b) procedure P_2

(c) procedure P_3 (d) procedure P_4

Fig. 4. Different visual border representations of vectors of two classes in MDF for the same simple monotone Boolean function

Fig. 3 in section 3 shows analysis of multivariate monotonicity of breast cancer cases that is almost perfect with four exclusions that are shown in red.

6 Conclusion and Future Work

Visual data mining had shown benefits in many areas. However, classical VDM methods do not address the specific needs for processing data that are highly overlapped in the visual space. This paper proposed a structural VDM approach and demonstrated its effectiveness in a medical application. To use this approach for Boolean vectors of a high dimensionality n we visualize only a part of the data structure that is actually needed for this.

We exploit monotonicity properties to build the *complete visual border* between two classes. This is an advantage relative to many traditional machine learning and data mining methods. The proposed method allows discovering the border between classes for a monotonized set of diagnostic features. In mathematical terms, this task is equivalent to the task known in discrete mathematics as *restoration of a monotone Boolean function*. The main idea of this process is based on decomposition of the binary cube E^n into Hansel chains that partition E^n without an overlap of chains.

It is proven a theorem that all elements with equal norms of adjacent Hansel chains 0C and 1C in E^{n+1} constructed from a single Hansel chain C in E^n have the same Hamming distance. It is also shown that for other adjacent Hansel chains this constant distance property holds in E^2-E^4, but it fails in E^5.

As a future research, it will be interesting to explore if another partitioning set of chains exists in E^n that satisfy the constant distance hypothesis for E^5 and for greater n.

Using the results of the mentioned theorem and associated statements we proposed an algorithm to locate Hansel chains on the Multiple Disk Form for visual data mining. The advantage of this algorithm is that it preserves similarity and monotonicity of n-D data visualized in 2-D or 3-D.

Multiple non-monotone data can be monotonized and then the monotonized data can be visualized by using the proposed technique. Future studies in monotonization include the decomposition of non-monotone Boolean functions and attributes into a combination of monotone Boolean functions. This study will require discovering limits of local monotonicity in data.

By further developing these procedures for non-monotone Boolean data and k-valued data structures, this approach can be used in a variety of applications including tasks, where data are dynamically changed or updated and the visual border between patterns also dynamically changes.

Acknowledgement

The authors are grateful to Karl Erich Wolff for multiple valuable comments and suggestions.

References

1. Beilken, C., Spenke, M.: Visual interactive data mining with InfoZoom -the Medical Data Set. In: The 3rd European Conference on Principles and Practice of Knowledge Discovery in Databases, PKDD (1999)
2. Birkhoff, G.: Lattice theory. AMS (1995)
3. Card, S.K., Mackinlay, J.: The structure of the information visualization design space. In: Proc. IEEE Symposium on Information Visualization, 92–99 (1997)
4. Ebert, D., Shaw, C., Zwa, A., Miller, E., Roberts, D.: Two-handed interactive stereoscopic visualization. In: IEEE Visualization 1996 Conference, pp. 205–210 (1996)
5. Hansel, G.: Sur le nombre des fonctions Booléennes monotones de n variables. C.R. Acad. Sci. Paris 262(20), 1088–1090 (1966)
6. Grätzer, G.: Lattice theory: first concepts and distributive lattices. W. H. Freeman and Co., New York (1971)
7. Groth, D., Robertson, E.: Architectural support for database visualization. In: Workshop on New Paradigms in Information Visualization and Manipulation, pp. 53–55 (1998)
8. Inselberg, A., Dimsdale, B.: Parallel coordinates: A tool for visualizing multidimensional Geometry. In: Proceedings of IEEE Visualization 1990, pp. 360–375. IEEE Computer Society Press, Los Alamitos (1990)
9. Keim, D., Ming, C.H., Dayal, U., Meichun, H.: Pixel bar charts: a visualization technique for very large multiattributes data sets. Information Visualization 1(1), 20–34 (2002)
10. Kovalerchuk, B., Triantaphyllou, E., Despande, A., Vityaev, E.: Interactive Learning of Monotone Boolean Functions. Information Sciences 94(1-4), 87–118 (1996)
11. Kovalerchuk, B., Vityaev, E., Ruiz, J.: Consistent and complete data and "expert" mining in medicine. In: Cios, K. (ed.) Medical Data Mining and Knowledge Discovery, pp. 238–280. Springer, Heidelberg (2001)

12. Kovalerchuk, B., Delizy, F.: Visual Data Mining using Monotone Boolean Functions. In: Kovalerchuk, B., Schwing, J. (eds.) Visual and Spatial Analysis, pp. 387–406. Springer, Heidelberg (2005)
13. Last, M., Kandel, A.: Automated perceptions in data mining, invited paper. In: IEEE International Fuzzy Systems Conference Proc. Part I, Seoul, Korea, pp. 190–197 (1999)
14. de Oliveira, M., Levkowitz, H.: From Visual Data Exploration to Visual Data Mining: A Survey. IEEE Trans. On Visualization and Computer Graphics 9(3), 378–394 (2003)
15. Peng, W., Ward, M., Rundensteiner, E.: Clutter Reduction in Multi-Dimensional Data Visualization Using Dimension Reordering. In: Proc. IEEE Information Visualization, pp. 89–96 (2004)
16. Post, F., van Walsum, T., Post, F., Silver, D.: Iconic techniques for feature visualization. In: Proceedings Visualization 1995, pp. 288–295 (1995)
17. Ribarsky, W., Ayers, E., Eble, J., Mukherja, S.: Glyphmaker: creating customized visualizations of complex data. IEEE Computer 27(7), 57–64 (1994)
18. Shaw, C., Hall, J., Blahut, C., Ebert, D., Roberts, A.: Using shape to visualize multivariate data. In: CIKM 1999 Workshop on New Paradigms in Information Visualization and Manipulation, pp. 17–20. ACM, New York (1999)
19. Thomas, J., Cook, K. (eds.): Illuminating the Path: The Research and Development, Agenda for Visual Analytics. National Visualization and Analytics Center (2005), http://nvac.pnl.gov/agenda.stm
20. Ward, M.: A taxonomy of glyph placement strategies for multidimensional data visualization. Information Visualization 1, 194–210 (2002)
21. Wilkinson, L., Anand, A., Grossman, R.: High-Dimensional Visual Analytics: Interactive Exploration Guided by Pairwise Views of Point Distributions. IEEE Transactions on Visualization and Computer Graphics 12(6), 1363–1372 (2006)
22. Di, Y., Rundensteiner, E., Ward, M.: Analysis Guided Visual Exploration of Multivariate Data. In: Proceedings of the IEEE Symposium on Visual Analytics, pp. 83–90 (2007)

Construction of an Event Tree on the Basis of Expert Knowledge and Time Series*

Gennady Lbov and Vladimir Berikov

Sobolev Institute of Mathematics,
Koptyug prosp. 4, 630090 Novosibirsk, Russia
{lbov,berikov}@math.nsc.ru
http://www.math.nsc.ru

Abstract. In this paper, we suggest an algorithm for event tree construction on the basis of an analysis of expert knowledge and multivariate time series. For the construction, we use the algorithm for dynamic objects recognition based on decision trees and the Bayesian pruning criterion.

Keywords: event tree, expert knowledge, decision tree, time series analysis.

1 Introduction

For the analysis of potentially dangerous events (technological catastrophes, extreme phenomena of nature etc) often models are used which have the form of event tree (also known as fault tree or error tree). Event trees (ET) allow to graphically represent the hierarchy of consecutively occurring stochastic events which cause certain undesirable event. On the basis of ET analysis, a forecast of undesirable events can be made, as well as measures for their prevention elaborated.

Usually an ET is designed by experts on the basis of their experience and other a priori information on the given event and on similar events. Often, the experts may use statistical information describing the object under investigation and similar objects, and this information can also be used for the construction of an event tree.

We assume that the problem in question has the following specific features:

– the features describing objects can be either binary, nominal or quantitative. Thus, we have a multidimensional series composed of a set of binary, symbolic, and numerical sequences;

– the probability distribution of the multidimensional random process is unknown. The decision function for the prediction is based on limited statistical material. Since the sample size is small and the space of states has high dimensionality, the derivation of statistically stable solutions becomes a problem;

* This work was supported by the Russian Foundation for Basic Research, projects 08-07-00136a, 10-01-00113a.

K.E. Wolff et al. (Eds.): KONT/KPP 2007, LNAI 6581, pp. 314–320, 2011.

– it can occur that some of the features are unmeasured at some moments. Thus, in the time series missing values may occur.

We suggest an algorithm for event tree design (or construction of some of its parts, for which it is necessary to analyze statistical data) by means of the analysis of multidimensional time series describing the dynamic properties of the object. As it is known, decision trees (DT) have a number of positive characteristics: form a simple hierarchical model of object, possess high generalization power in case of small sample size, give a possibility to work with both quantitative and qualitative features, with data of low quality (having missing values etc). Thus the implementation of DTs in event tree generation looks promising.

2 Event Tree and Decision Tree

The root of ET (fig. 1) corresponds to the so called top undesirable event (TUE). The nodes of the first layer of ET correspond to some events, they precede the TUE and they are able to cause it. These events can be connected by gates AND, OR. Besides that, the probabilities with which the specified events cause TUE are given. Similarly, each node (event) of the first layer of the tree can be matched to nodes (events) of the second layer etc. The leaves of the tree denote the events that are impossible to decompose due to the absence of information or other reasons.

It is required to find a dependency model of some feature Y from features X. Such a model can be represented in the form of a decision tree [1,2]. The decision tree (DT) consists of nodes and branches connecting the nodes. Each node corresponds to a certain feature and the branch corresponds to a range of values. When precisely two branches grow out from an internal node (the tree of such type is called a dichotomous tree), each of these branches corresponds to a true or false statement concerning the given feature. The value Y is ascribed for each terminal node of a tree ("leaf"). In case of a pattern recognition problem

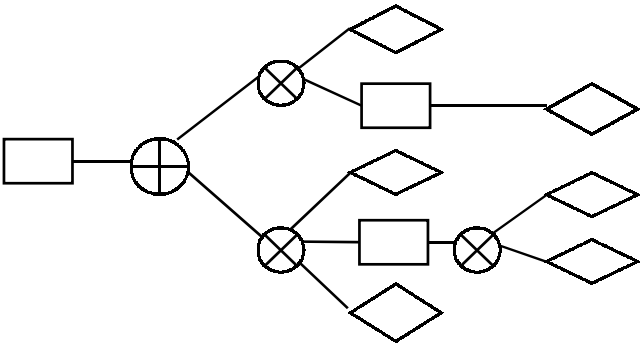


Fig. 1. Event tree. □ - events; ♢ - leaves; ⊕ - gate "OR"; ⊗ - gate "AND".

this value is a certain class (pattern). For any observation x, using DT, we can find the predicted value Y: the value Y_S ascribed to S-th leaf will be the forecast for x.

3 Multidimensional Time Series and Decision Trees

Let random features $X(t) = (X_1(t),\ldots, X_n(t))$, whose values vary with time, be used to describe a certain object. We denote the set of possible values of X_j by D_j. The following types of features can be distinguished:

– binary feature: D_j= {0,1};

– nominal feature: D_j=$\{u_j^1,\ldots,u_j^{l_j}\}$, where $\{u_j^1,\ldots,u_j^{l_j}\}$ is a set of symbols (names);

– quantitative feature: $D_j \subseteq R$, where R is a set of real numbers.

The binary and nominal features are also referred to as qualitative features.

Let features be measured at the consecutive moments of time $t^1,\ldots,t^\mu,\ldots$. For definiteness we will assume that measurements are carried out through equal intervals of time. We will designate through $x_j(t^\mu) = X_j(t^\mu)$ the value of feature X_j at the moment of time t^μ. Thus, we have a n-dimensional heterogeneous time series $x_j(t^\mu), j = 1,\ldots,n,\ \mu = 1,2,\ldots$.

Let us choose one predicted feature X_{j_0}, $1 \leqslant j_0 \leqslant n$. We designate, for convenience, this feature through Y. Let us consider the moment of time t^μ, and also a set of previous moments of time $t^{\mu-1}, t^{\mu-2},\ldots,t^{\mu-l}$, where l is a given size ("deep of history"), $1 \leqslant l < \mu$.

We suppose that the conditional distribution of $Y(t^\mu)$, when all previous values $X(t)$ are given, depends only on the values of the series in l previous moments of time.

Besides, we suppose that this dependence is the same for any value μ. The assumption means that the statistical properties of series determining dependence are stationary.

For any moment of time t^μ it is possible to form a series $v^\mu = (X_j(t^{\mu-i}))$, $i = 1,\ldots,l,\ j = 1,\ldots,n$, representing the time series in l previous moments of time. We will call a series v^μ the background of length l for the moment t^μ.

It is required to construct a model of dependence of feature Y from its background for any moment of time. The model allows to predict the value of feature Y at a future moment of time from the historical values of features for the l last moments. In other words, the given model, using background, represents a decision function for forecasting. In analogy to a usual pattern recognition problem, we will call a problem of the given type a problem of recognition of a dynamic object. The analyzed object can change its class in the run of time.

We will represent a decision function for forecasting time series on its background as a decision tree. This decision tree differs from usual tree in its nodes, each of them corresponds to a feature X_j in some i-th moment of past time. For convenience, we will designate these features, with a glance to background, through X_j^i. Thus, X_j^i means feature X_j in i-th previous moment of time (concerning a present situation).

Let there be a set of measurements of features $X = (X_1, \ldots, X_n)$ at the moments of time $t^1, \ldots, t^N$ and value l is also given. Thus, we have a multivariate heterogeneous time series of length N. We generate the set of all histories of length l for the moments of time $t^{l+1}, \ldots, t^N$: $A = v^\mu$, $\mu = l+1, \ldots, N$.

For any given decision tree for forecasting by background (DTFB), it is possible to define its quality: we will designate through $\hat{Y}(t^\mu)$ the predicted value of Y received with the help of the tree by the background v^μ. The criterion of quality will be

$$Q = \frac{1}{N-l} \sum_{\mu=l+1}^{N} h(\mu),$$

where $h(\mu) = L(\hat{Y}(t^\mu), Y(t^\mu))$, $L(a, b)$ be the loss function determining losses if class a is predicted then b is the true class. Thus, Q is the estimate of the risk function.

The initial problem of constructing the DTFB is divided into simpler pattern recognition problems. We will present series v^μ as the table $v^\mu = (X_j(t^{\mu-i}), i = 1, \ldots, l, j = 1, \ldots, n$ containing l rows and n columns. Then, the initial information for forecasting is the set of tables v^μ, together with the values of predicted feature Y specified for each table $Y(t^\mu)$, $\mu = l+1, \ldots, N$. It is possible to present the set $A = v^{l+1}, \ldots, v^N$ as a 3-dimensional table of dimension $l \times n \times (N-l)$ to which the vector $(y^{l+1}, \ldots, y^N)$ corresponds. However, available methods of recognition based on DT use 2-dimensional tables as input information. Below we suggest an alternative way to use the given methods for the analysis of 3-dimensional data tables. Consider l tables $X_j(t^{\mu-i}), Y(t^\mu)$, where $j \in \{1, 2, \ldots, n\}$, $\mu \in \{l+1, l+2, \ldots, N\}$, $i \in \{1, 2, \ldots, l\}$. Thus, we have l horizontal cuts of a 3-dimensional data table. For each cut (2-dimensional table), the decision tree minimizing Q is constructed (see [5] for details). As a result, we receive a set of trees $T_1, T_2, \ldots, T_l$. We will denote the best of these trees (with minimal Q) as T^*.

4 From Decision Trees to Event Trees

Let feature Y correspond to the occurrence of some undesirable event: Y=1, if this event occurs, Y=0 otherwise. Thus, the number of predicted classes equals two. Consider a loss function with the following properties: $L(0,0) = L(1,1) = 0$; $L(1,0) << L(0,1)$ (the error in classifying an undesirable event as an ordinary event costs much more than the error in classifying an ordinary event as an undesirable event).

In T^*, all features are measured for the same moment of time in the past. This tree is then used for the formation of the first layer of an ET. Consider the following algorithm:

1. Find all statements in paths from root node of T^* to the leaves that correspond to undesirable events;

2. Use these statements as nodes (events) in the current level of ET (see an example on fig. 2). The conjunctions of statements are connected with TUE by

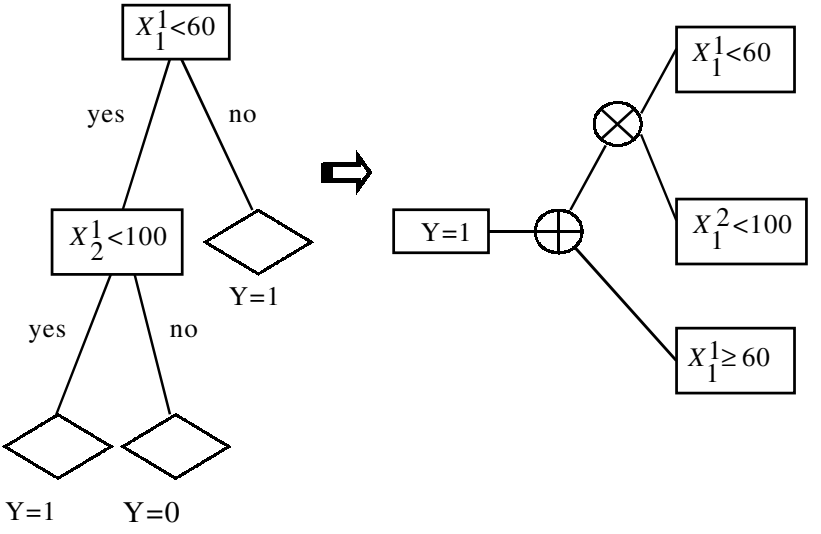


Fig. 2. Construction of an event tree on the basis of a decision tree

means of AND gates, and different paths are connected with TUE by means of OR gates. Choose non-terminal nodes of ET;

3. For each non-terminal event E_m of the current level of ET form a new feature Z_m: $Z_m = 1$, if this event takes place, $Z_m = 0$ otherwise, where $m = 1, \ldots, M$, M is the number of non-terminal nodes;

4. For each m construct a sub-tree for the prediction of Z_m on the basis of its background;

5. Use the statements of the obtained DTFB as nodes (events) for the next level of ET;

6. Repeat steps 3-5 until no non-terminal nodes left.

The constructed ET may be corrected and complemented by an expert. The probabilities of events in ET are evaluated as frequencies of corresponding observations in time series.

5 Bayesian Criteria for Decision Tree Pruning

Often the pruning methods based on the division of learning samples into two parts are used for decision tree design. The first part is used to grow a decision tree which is probably overtrained and has a large number of leaves. The second part is used to prune this decision tree with the purpose to improve its recognition performance. The pruning methods are described, for example, in [3].

By numbering the leaves of a tree, we can reduce the problem to one feature X. The values of this feature ("cells") are coded by numbers $1, \ldots, m, \ldots, M$, where M is the number of leaves. Let $p_m^{(q)}$ be the probability of the joint event "$X = m, Y = q$". Denote the a priori probability of the i-th class as $p^{(i)}$. It is evident

that $\sum_q p^{(q)} = 1$, $\sum_m p_m^{(q)} = p^{(q)}$, $q = 1, 2$. Let N be the pruning sample size, $n_m^{(q)}$ be the frequency with which the observations of the q-th class fall into the m-th cell. Denote $s = (n_1^{(1)}, \ldots, n_M^{(2)})$. Let us consider the family of distribution models with a set of parameters $\Theta_p = \{\theta\}$, where $\theta = (p_1^{(1)}, \ldots, p_M^{(2)})$, $m = 1, \ldots, M$. The random vector of frequencies s corresponds to a multynomial distribution with parameter vector θ. In applied problems of recognition the vector θ is usually unknown. We use the Bayesian approach: suppose that the random vector Θ has a known a priori distribution $p(\theta)$. In the given work the case of uniform density $p(\theta) = const$ is considered. This assumption is appropriate, then a priori vagueness in choice of model takes place. Let $Y = f(m)$ be a decision function mapping each leaf to some class. In [4], the following theorem was proven:

Theorem 1. *A posteriori mathematical expectation of the risk for a decision function f and given a priori probabilities of classes equals*

$$R_{f,s} = \sum_m \sum_q p^{(q)} L(f(m), q) \frac{n_m^{(q)} + 1}{n^{(q)} + M},$$

where $n^{(q)}$ is the number of examples of q-th class in pruning sample.

The value $R_{f,s}$ will be called the Bayesian estimate of risk for the decision function f, sample s and known priors. This value can be used as a criterion of quality for different variants of a pruned tree. The following greedy algorithm of pruning is applied.

1. For each node of the initial tree compute the Bayesian estimate of risk for the tree in which this node is pruned.
2. Choose the variant of pruning with the smallest estimate of risk.
3. Repeat steps 1,2 for the selected variant until the estimate of risk stops improving.

6 Experiments

To test the suggested algorithm, the following numerical experiment was done. Two random Boolean and one random numerical sequences were generated independently. The length of the sequences equals 1000. It was set that for the defined combination of previous values of these sequences the undesirable event should take place (with probability 0,75), but in all other cases this event did not take place. Thus, the Boolean sequence denoting the presence or absence of the undesirable event was formed (the frequency of this event was about 0,09). To study the behavior of the algorithm in situations when some of the features values are unknown, it was additionally assumed that 5% of the learning sequences have missing values (the places of missing values were taken at random). The sequences were analyzed with the suggested algorithm.

The recursive algorithm ("R-method" [6]) was used for tree growing. For quality estimation, 10-fold cross-validation technique was applied. As a result, the test sequence having 100 situations describing undesirable events was recognized

with one error. Cross-validation risk estimate was 0,086. In the same time, the standard method of pruning based on empirical error criterion gave an estimate of risk which was 7% larger.

In the next experiment, the applied problem of low water-level prediction of the river Ob (in the Altai region of Russia) was analyzed. The data set included per-month measurements of temperature, precipitation and water expenditures from 1922 to 2000. There were 7 extreme low-water situations in winter months during this period. This sequence was analyzed with the suggested algorithm. Firstly the whole sequence was taken for the analysis; the risk estimate showed an unacceptable value (for quality estimation, one-hold-out method was used). Hypothetically it can be explained by the global rise in temperature on the Earth in last decades. For checking this hypothesis, two rows were formed from the data set: for the first years of observing, and for the last 25 years. After that, the results became more reasonable: 0,14 error rate for extreme situations (i.e. only one low water-level situation was erroneously predicted from seven situations) and 0,26 for ordinary events.

7 Summary

In the given work we suggest an algorithm for event tree design by means of the analysis of expert knowledge and multivariate time series describing the dynamic properties of an object. The algorithm uses the Bayesian criterion for decision tree pruning, and can be used for the analysis of extreme events in the conditions of high a priori uncertainty about them. In this case an expert can contribute additional statistical information concerning the object under investigation. This information is then used for the formation of event trees by means of decision trees. The experiments with the artificial and real data sets confirm the usefulness of the algorithm.

References

1. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth Int. Group, Belmont (1984)
2. Berikov, V.B., Lbov, G.S.: Analysis of statistical data with use of decisions trees, `http://www.math.nsc.ru/AP/datamine/eng/decisiontree.htm`
3. Esposito, F., Malerba, D., Semerato, G.: A comparative analysis of methods for pruning decision trees. IEEE Trans. Pattern Anal. And Machine Intelligence 19(5), 476–491 (1997)
4. Berikov, V.B.: A Priori Estimates of Recognition Accuracy for a Small Training Sample Size. Computational Mathematics and Mathematical Physics 43(9), 1377–1386 (2003)
5. Lbov, G.S., Berikov, V.B.: Recognition of a Dynamic Object and Prediction of Quantitative Characteristics in the Class of Logical Functions. Pattern Recognition and Image Analysis 7(4), 407–413 (1997)
6. Lbov, G.S., Berikov, V.B.: Recursive Method of Formation of the Recognition Decision Rule in the Class of Logical Functions. Pattern Recognition and Image Analysis 3(4), 428–431 (1993)

Author Index