A Prolog-like Inference System for Computing Minimum-Cost Abductive Explanations in Natural-Language Interpretation*

(Running Title: Prolog-like System for Abductive Explanations)

Mark E. Stickel

Artificial Intelligence Center

SRI International

333 Ravenswood Avenue

Menlo Park, California 94025

*This research is supported by the Defense Advanced Research Projects Agency, under Contract N00014-85-C-0013 with the Office of Naval Research, and by the National Science Foundation, under Grant CCR-8611116. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency, the National Science Foundation, or the United States government. Approved for public release. Distribution unlimited.

Abstract

By determining what added assumptions would suffice to make the logical form of a sentence in natural language provable, abductive inference can be used in the interpretation of sentences to determine what information should be added to the listener's knowledge, i.e., what he should learn from the sentence. This is a comparatively new application of mechanized abduction. A new form of abduction—least specific abduction—is proposed as being more appropriate to the task of interpreting natural language than the forms that have been used in the traditional diagnostic and design-synthesis applications of abduction. The assignment of numerical costs to axioms and assumable literals permits specification of preferences on different abductive explanations. A new Prolog-like inference system that computes abductive explanations and their costs is given. To facilitate the computation of minimum-cost explanations, the inference system, unlike others such as Prolog, is designed to avoid the repeated use of the same instance of an axiom or assumption.

1 Introduction

We introduce a Prolog-like inference system for computing minimum-cost abductive explanations. This work is being applied to the task of natural-language interpretation, but other applications abound. Abductive inference is inference to the best explanation. The process of interpreting sentences in discourse can be viewed as the process of generating the best explanation as to why a sentence is true, given what is already known [9]—that is, determining what information must be added to the listener's knowledge (what assumptions must be made) for him to know the sentence to be true.[1]

To appreciate the value of an abductive inference system over and above that of a merely deductive inference system, consider a Prolog specification of graduation requirements (e.g., to graduate with a computer science degree, one must fulfill the computer science, mathe-

[1] Alternative abductive approaches to natural-language interpretation have been proposed by Charniak [3] and Norvig [11].

matics, and engineering requirements; the computer science requirements can be satisfied by taking certain courses, etc.) as an example of a deductive-database application [10]:

```
csReq <- basicCS, mathReq, advancedCS, engReq, natSciReq.
engReq <- digSys.
natSciReq <- physicsI, physicsII.
natSciReq <- chemI, chemII.
natSciReq <- bioI, bioII.
     :
```

After adding facts about which courses a student has taken, such a database can be queried to ascertain whether the student meets the requirements for graduation. Evaluating `csReq` in Prolog will result in a yes or no answer. However, standard Prolog deduction cannot determine what more must be done to meet the requirements if they have not already been fulfilled; that would require analysis to find out why the deduction of `csReq` failed.

This sort of task can be accomplished by abductive reasoning. Given what is known in regard to which courses have been taken, what assumptions could be made to render provable the statement that all graduation requirements have been met?

2 Three Abduction Schemes

We will consider here the abductive explanation of conjunctions of positive literals from Horn clause knowledge bases. An explanation will consist of a substitution for variables in the conjunction and a set of literals to be assumed. In short, we are developing an abductive extension of pure Prolog.

The general approach can be characterized as follows: when trying to explain why $Q(a)$ is true, hypothesize $P(a)$ if $P(x) \supset Q(x)$ is known.

The requirement that assumptions be literals does not permit us to explain $Q(a)$ when $P(a)$ is known by assuming $P(x) \supset Q(x)$, or even $P(a) \supset Q(a)$. We do not regard this as a limitation in tasks like diagnosis and natural-language interpretation. Some other tasks, such as scientific-theory formation, could be cast in terms of abductive explanation when the assumptions take these more general forms.

We want to include the possibility that $Q(a)$ can be explained by assuming $Q(a)$. As later examples will show, this is vital in the natural-language interpretation task.

Consider again the example of the deductive database for graduation requirements. All the possible ways of fulfilling the requirements can be obtained by backward chaining from `csReq`:

```
<- csReq.
<- basicCS, mathReq, advancedCS, engReq, natSciReq.
<- basicCS, mathReq, advancedCS, engReq, physicsI, physicsII.
<- basicCS, mathReq, advancedCS, engReq, chemI, chemII.
<- basicCS, mathReq, advancedCS, engReq, bioI, bioII.
<- basicCS, mathReq, advancedCS, digSys, natSciReq.
<- basicCS, mathReq, advancedCS, digSys, physicsI, physicsII.
<- basicCS, mathReq, advancedCS, digSys, chemI, chemII.
<- basicCS, mathReq, advancedCS, digSys, bioI, bioII.
      :
```

Eliminating from any such clause those requirements that have been met results in a list that, if met, would result in fulfilling the graduation requirements. Different clauses can be more or less specific about how the remaining requirements must be satisfied. If the student lacks only Physics II to graduate, the statements that he can fulfill the requirements for graduation by satisfying `physicsII`, `natSciReq`, or (rather uninformatively) `csReq` can all be derived by this backward-chaining scheme.

The above clauses are all possible abductive explanations for the graduation requirements' being met.

In general, if the formula $Q_1 \wedge \cdots \wedge Q_n$ is to be explained or abductively proved, the substitution [of values for variables] θ and the assumptions $P_1, \ldots, P_m$ would constitute one possible explanation if $(P_1 \wedge \cdots \wedge P_m) \supset (Q_1 \wedge \cdots \wedge Q_n)\theta$ is a consequence of the knowledge base.

If, in the foregoing example, the student lacks only Physics II to graduate, assuming `physicsII` then makes `csReq` provable.

If the explanation contains variables (for example, if $P(x)$ is an assumption used to explain $Q(x)$), the explanation should be interpreted as neither to assume $P(x)$ for all x (i.e., assume $\forall x P(x)$) nor to assume $P(x)$ for some unspecified x (i.e., assume $\exists x P(x)$), but rather that, for any variable-free instance t of x, if $P(t)$ is assumed, then $Q(t)$ follows.

It is a general requirement that the conjunction of all the assumptions made be consistent with the knowledge base. (In the natural-language interpretation task, the validity of rejecting assumptions that are inconsistent with the knowledge base presupposes that the knowledge base is correct and that the speaker of the sentence is neither mistaken nor lying.)

Prolog-style backward chaining, with an added factoring operation and without the literal ordering restriction (so that any, not just the leftmost, literal of a clause can be resolved on), is capable of generating all possible explanations that are consistent with the knowledge base. That is, every possible explanation consistent with the knowledge base is subsumed by an explanation that is generable by backward chaining and factoring.

It would be desirable if the procedure were guaranteed to generate no explanations that are inconsistent with the knowledge base. However, this is impossible; consistency of explanations with the knowledge base must be checked outside the abductive-reasoning inference system. (Not all inconsistent explanations are generated: the system can generate only those explanations that assume literals that can be reached from the initial formula by backward chaining.) Determining consistency is undecidable in general, though decidable subcases do exist, and many explanations can be rejected quickly for being inconsistent with the knowledge base. For example, assumptions can be readily rejected if they violate sort or ordering restrictions, e.g., assuming $woman(John)$ can be disallowed if $man(John)$ is known or already assumed, and assuming $b < a$ can be disallowed if $a \leq b$ is known or already assumed. Sort restrictions are particularly effective in eliminating inconsistent explanations in natural-language interpretation. We shall not discuss the consistency requirement further; what we are primarily concerned with here is the process of generating possible explanations, in order of preference according to our cost criteria, not with the extra task of verifying their consistency with the knowledge base.

Obviously, *any* clause derived by backward chaining and factoring can be used as the list of assumptions to prove the correspondingly instantiated original clause abductively. This can result in an overwhelming number of possible explanations. Various abductive schemes have been developed to limit the number of acceptable explanations.

What we shall call *most specific abduction* has been used particularly in diagnostic tasks. In explaining symptoms in a diagnostic task, the objective is to identify causes that, if assumed to exist, would result in the symptoms. The most specific causes are usually sought, since identifying less specific causes may not be as useful.

What we shall call *predicate specific abduction* has been used particularly in planning and design-synthesis tasks. In generating a plan or design by specifying its objectives and ascertaining what assumptions must be made to make the objectives provable, acceptable assumptions are often expressed in terms of a prespecified set of predicates. In planning, for example, these might represent the set of executable actions.

We consider what we will call *least specific abduction* to be especially well suited to natural-language-interpretation tasks. Given that abductive reasoning has been used mostly for diagnosis and planning, and that least specific abduction tends to produce what would be considered frivolous results for such tasks, least specific abduction has been little studied. Least specific abduction is used in natural-language interpretation to seek the least specific assumptions that explain a sentence. More specific explanations would unnecessarily and often incorrectly make excessively detailed assumptions.

2.1 Most Specific Abduction

Resolution-based systems for abductive reasoning applied to diagnostic tasks [12, 5, 6] have favored most specific explanations by stipulating that only *pure* literals (those that cannot be resolved with any clause in the knowledge base) be adoptable as assumptions. Explanations can be constructed from *dead-end* [5, 6] clauses obtained by backward-chaining deduction from the formula to be explained, where a dead-end clause is one composed entirely of pure literals. For causal-reasoning tasks, this eliminates frivolous and unhelpful explanations for "the watch is broken" such as simply noting that the watch is broken, as opposed to, perhaps, the mainspring's being broken. The explanations can be too specific. In diagnosing the failure of a computer system, most specific abduction could never merely report the failure of a board if the knowledge base has enough information for the board's failure to be explained—possibly in many alternative, inconsistent ways—by the failure of its components.

Besides sometimes providing overly specific explanations (discussed further in Section 2.3), most specific abduction is incomplete—it does not compute all the reasonable most specific explanations.

Consider explaining instances of the formula $P(x) \wedge Q(x)$ with a knowledge base that consists of $P(a)$ and $Q(b)$. Most specific abduction's backward chaining to sets of pure literals makes $P(c) \wedge Q(c)$ explainable by assuming $P(c)$ and $Q(c)$ (both literals are pure), but $P(x) \wedge Q(x)$ is explainable only by assuming $P(b)$ or $Q(a)$, since $P(x)$ and $Q(x)$ are not pure. The explanation that assumes $P(c)$ and $Q(c)$, or any value of x other than a or b, to explain $P(x) \wedge Q(x)$ will not be found.

Thus, most specific abduction does not "lift" properly from the case of ground (variable-free) formulas to the general case (this would not be a problem if we restricted ourselves to propositional-calculus formulas). A solution would be to require that all generalizations of any pure literal also be pure. This too is often impractical, since purity of $P(c)$ in the above example would require purity of $P(x)$, which is inconsistent with the presence of $P(a)$ in the knowledge base.

A special case of the requirement that generalizations of pure literals be pure would be to have a set of predicates that do not occur positively (i.e., they appear only in negated literals) in the knowledge base. But the case of a set of assumable predicate symbols is handled more generally, i.e., without the purity requirement, by predicate specific abduction (see Section 2.2). This is consistent with much of the practice in diagnostic tasks, where causal explanations in terms of particular predicates, such as Ab, are often sought.

2.2 Predicate Specific Abduction

Resolution-based systems for abductive reasoning applied to design-synthesis and planning tasks [7] have favored explanations that are expressed in terms of a prespecified subset of the predicates, namely, the assumable predicates.

With a knowledge base that consists of $P(a)$ and $Q(b)$, predicate specific abduction could explain $P(x) \wedge Q(x)$ being true by (1) assuming $P(b)$, provided P is an assumable predicate, (2) assuming $Q(a)$, provided Q is an assumable predicate, along with (3) assuming $P(x) \wedge Q(x)$, provided both P and Q are assumable predicates.

2.3 Least Specific Abduction

The criterion for "best explanation" that must be applied in natural-language interpretation differs greatly from most specific abduction for diagnostic tasks. To interpret the sentence "the watch is broken," the conclusion will likely be that we should add to our knowledge the information that the watch (i.e., the one currently being discussed) is broken. The explanation that would be frivolous and unhelpful in a diagnostic task is just right for sentence interpretation. A more specific causal explanation, such as the mainspring's being broken, would be gratuitous.

Associating the assumability of a literal with its purity as most specific abduction does yields not only causally specific explanations, but also taxonomically specific explanations. With axioms like $mercury(x) \supset liquid(x)$, $water(x) \supset liquid(x)$, explaining $liquid(a)$, when $liquid(a)$ cannot be proved, would require the assumption that a was mercury, or that it was water, and so on. Not only are these explanations more specific than the only fully warranted one that a is simply a liquid, but none may be correct, for example, if a is actually milk, but milk is not mentioned as a possible liquid. Most specific abduction thus assumes completeness of the knowledge base with respect to causes, subtypes, and so on. The purity requirement may make it impossible to make any assumption at all. Many reasonable axiom sets contain axioms that make literals, which we would sometimes like to assume, impure and unassumable. For example, in the presence of $parent(x, y) \supset child(y, x)$

and $child(x, y) \supset parent(y, x)$, neither $child(a, b)$ nor $parent(b, a)$ could be assumed, since neither literal is pure.

We note that assuming any literals other than those in the original formula generally results in more specific (and thus more likely to be wrong and riskier) assumptions. When explaining R with $P \supset R$ (or $P \wedge Q \supset R$) in the knowledge base, either R or P (or P and Q) can be assumed to explain R. Assumption of R, the consequent of an implication, in preference to antecedent P (or P and Q), results in the fewest consequences. Assuming the antecedent may result in more consequences, e.g., if other rules like $P \supset S$ are present.

Predicate specific abduction is not ideal for natural-language interpretation either, since there is no easy division of predicates into assumable and nonassumable ones so that those assumptions that can be made will be reasonably restricted. Most predicates must be assumable in some circumstances, e.g., when certain sentences are being interpreted, but in many other cases should not be assumed.

Least specific abduction, wherein a subset of the literals asked to be proven must be assumed, comes closer to our ideal of the right method of explanation for natural-language interpretation. Under this model, a sentence is translated into a logical form that contains literals whose predicates stand for properties and relationships and whose variable and constant arguments refer to entities specified or implied by the sentence. The logical form is then proved abductively, with some or all of the variable values filled in from the knowledge base and unprovable literals of the logical form assumed.

The motivation for this is the claim that what we should learn from a sentence is often near the surface and can attained by assuming literals in the sentence's logical form. For example, when interpreting

The car is red.

with logical form

$$car(x) \wedge red(x),$$

we would typically want to ascertain from the discourse which car x is being discussed and learn by abductive assumption that it is red and not something more specific, such as the fact that it is carmine or belongs to a fire chief (whose cars, according to the knowledge base, might always be red).

3 Assumption Costs

A key issue in abductive reasoning is picking the *best* explanation. Which one is indeed best is so subjective and task-dependent that there is no hope of devising an algorithm that will always compute [only] the best explanation. Nevertheless, there are often so many abductive explanations that it is necessary to have some means of eliminating most of them. We attach numerical assumption costs to assumable literals and compute minimum-cost abductive explanations in an effort to influence the abductive reasoning system into favoring the intended explanations.

We regard the assignment of numerical costs as a part of programming the explanation task. The values used may be determined by subjective estimates of the likelihood of various interpretations or perhaps they may be learned through exposure to a large set of examples. A minimum-cost scheme similar to ours, which includes costs on antecedent literals in implications as described in Section 5.1, can be ascribed a probabilistic interpretation [4]. In our experience, preferences among explanations are relatively insensitive to the exact values of the costs.

In selecting the best abductive explanation, we often prefer, when given the choice, that certain literals be assumed rather than others. For example, when the sentence

The car is red.

with the logical form

$$car(x) \wedge red(x)$$

is being interpreted, the knowledge base will likely contain both cars and things that are red. However, the form of the sentence suggests that $red(x)$ is new information to be learned and that $car(x)$ should be proved from the knowledge base because it is derived from a definite reference, i.e., a specific car is presumably being discussed. Thus, an explanation that assumes $red(a)$ where $car(a)$ is provable should be preferred to an explanation that assumes $car(b)$ where $red(b)$ is provable. A way to express this preference is through numerical assumption costs associated with the assumable literals: $car(x)$ could have cost 10, and $red(x)$ cost 1. Alternatively, a logical form that insisted upon proving $car(x)$ and assuming $red(x)$ might have been used. We prefer this more neutral logical form to allow the possibility of alternative interpretations, which may be weighted against but not absolutely ruled out by the cost assignment.

The cost of an abductive explanation could then just be the sum of the assumption costs of all the literals that had to be assumed: $car(a) \wedge red(a)$ would be the preferred explanation, with cost 1, and $car(b) \wedge red(b)$ would be another explanation, with higher cost 10.

However, if only the cost of assuming literals is counted in the cost of an explanation, there is in general no effective procedure for computing a minimum-cost explanation. For example, if we are to explain P, where P is assumable with cost 10, then assuming P produces an explanation with cost 10, but proving P would result in a better explanation with cost 0. Since provability of first-order formulas is undecidable in general, it may be impossible to determine whether the cost 10 explanation is best.

The solution to this difficulty is that the cost of proving literals, as well as the cost of assuming them, must be included in the cost of an explanation. An explanation that assumes P with cost 10 would be preferred to an explanation that proves P with cost 50 (e.g., in a proof of 50 steps) but would be rejected in favor of an explanation that proves P with cost less than 10.

Although treating explanation costs as composed only of assumption costs is conceptually elegant (why should we distinguish explanations that differ in the size of their proof, when only their provability should matter?), there are substantial advantages gained by taking into account proof costs as well as assumption costs, in addition to the crucial benefit of making the search for a minimum-cost explanation theoretically possible.

If costs are associated with the axioms in the knowledge base as well as with assumable literals, these costs can be used to encode information on the likely relevance of the fact or rule to the situation in which the sentence is being interpreted.

Axiom costs can be adjusted to reflect the salience of certain facts. If a is a car mentioned in the previous sentence, the cost of the axiom $car(a)$ could have been adjusted downward so that the explanation of $car(x) \wedge red(x)$ that assumes $red(a)$ would be preferred to one that assumes $red(c)$ for some other car c in the knowledge base.

Indeed, the explanation that assumes $red(a)$ should probably be preferred to any explanation that proves both $car(c)$ and $red(c)$ (i.e., there is a red car in the knowledge base—this would be a "perfect" zero-cost explanation if only assumption costs were used), since the recent mention of a makes it likely that a is the subject of the sentence and that the purpose of the sentence is to convey the new information that a car is red—interpreting the referent of "the car" as a car that is already known to be red results in no new information being learned.

We have some reservations about choosing explanations on the basis of numerical costs. Nonnumerical specification of preferences is an important research topic. Nevertheless, we have found these numerical costs to be quite practical. Numerical costs offer an easy way of specifying that one literal is to be assumed rather than another.

When many alternative explanations are possible, the summation of numerical costs in each explanation and the adoption of an explanation with minimum total cost provides a mechanism for trading off the costs of one proof and set of assumptions against the costs of another. If this method of comparing explanations is too simple, other means may be too complex to be realizable, since they would require preference choices among a wide variety of sets of assumptions and proofs.

We provide a procedure for computing a minimum-cost explanation by enumerating possible partial explanations in order of increasing cost. Even a perfect scheme for specifying preferences among alternative explanations may not lead to an effective procedure for generating a most preferred one, as there may be no way of cutting off the search for an explanation with the certainty that the best explanation exists among those so far discovered. Finally, any scheme will be imperfect: people may disagree as to the best explanation of some data and, moreover, sometimes do misinterpret sentences.

4 Minimum-Cost Proofs

We now present the inference system for computing abductive explanations. This method applies to both predicate specific and least specific abduction. We have not tried to incorporate most specific abduction into this scheme because of its incompleteness, its incompatibility with ordering restrictions, and its unsuitability for natural-language interpretation. In predicate specific abduction, the assumability of a literal is determined by its predicate symbol and assumption costs are specified on a predicate-by-predicate basis. In least specific abduction, only literals in the formula to be explained are assumable, and their assumption costs are directly associated with them.

The cost of a proof is usually taken to be a measure on the syntactic form of the proof, e.g., the number of steps in the proof. A more abstract characterization of cost is called for. We want to assign different costs to different inferences by associating costs with individual axioms; we also want to have a cost measure that is not so dependent on the syntactic form of the proof.

We assign to each axiom A a cost $cost(A)$ that is greater than zero. Likewise we assign a cost $cost(A)$ greater than zero to each assumable literal A. When looked at abstractly, a proof is a demonstration that the goal follows from a set S of substitution instances of the axioms, together with, in the case of abductive proofs, a set H of substitution instances of assumable literals that are assumed in the proof. We want to count the cost of each separate instance of an axiom or assumption only once instead of the number of times it may appear in the syntactic form of the proof. Thus, a natural measure of the cost of the proof is

$$\sum_{A\sigma \in S} cost(A) + \sum_{A\sigma \in H} cost(A)$$

Consider the example of explaining $Q(x) \wedge R(x) \wedge S(x)$ with a knowledge base that includes $P(a)$, $P(x) \supset Q(x)$, and $Q(x) \wedge R(x) \supset S(x)$ and with R being assumable by using Prolog plus an inference rule for assuming literals:

```
1. <- Q(x), R(x), S(x).
2. <- P(x), R(x), S(x).      % resolve 1 with Q(x) <- P(x)
3. <- R(a), S(a).            % resolve 2 with P(a)
4. <- S(a).                  % assume R(a) in 3
5. <- Q(a), R(a).            % resolve 4 with S(x) <- Q(x), R(x)
6. <- P(a), R(a).            % resolve 5 with Q(x) <- P(x)
7. <- R(a)                   % resolve 6 with P(a)
8. <- true                   % assume R(a) in 7
```

$Q(x) \wedge R(x) \wedge S(x)$ has been explained with x having the value a under the assumption that $R(a)$ is true.

The cost of the proof is the sum of the costs of the axiom instances $P(a)$, $P(a) \supset Q(a)$, and $Q(a) \wedge R(a) \supset S(a)$, plus the cost of assuming $R(a)$. The costs of using $P(a)$ and $P(x) \supset Q(x)$ and assuming $R(a)$ are not counted twice even though they were used twice, since the same instances were used or assumed. If we had had occasion to use $P(x) \supset Q(x)$ with b as well as a substituted for x, then the cost of $P(x) \wedge Q(x)$ would have been added in twice.

In general, the cost of a proof can be determined by extracting the sets of axiom instances S and assumptions H from the proof tree and performing the above computation. However, it is an enormous convenience if there always exists a *simple proof tree* such that each separate instance of an axiom or assumption actually occurs only once in the proof tree. That way, as the inferences are performed, costs can simply be added to compute the cost of the current partial proof. (Even if the same instance of an axiom or assumption happens to be used and counted twice, a different, cheaper derivation would use and count it only once.) Partial proofs can be enumerated in order of increasing cost by employing breadth-first or iterative-deepening search methods and minimum-cost explanations can be discovered effectively. Iterative-deepening search is compatible with maintaining Prolog-style implementation and performance [15, 16].

We shall describe our inference system as an extension of pure Prolog. Prolog, though complete for Horn sets of clauses, lacks this very desirable property of always being able to find a simple proof tree.

Prolog's inference system—ordered input resolution without factoring—would have to both eliminate the ordering restriction and add the factoring operation to remain a form of resolution and be able to prove $\leftarrow Q, R$ from $Q \leftarrow P$, $R \leftarrow P$, and P without using P twice. Elimination of the ordering restriction is potentially very expensive. For example, there are $n!$ proofs of $\leftarrow Q_1, \ldots, Q_n$ from the axioms $Q_1, \ldots, Q_n$ when unordered input resolution is used, but only one with ordered input resolution. (Most specific abduction performs unordered input resolution [12, 5, 6].)

We present a resolution-like inference system, an extension of pure Prolog, that preserves the ordering restriction and does not require repeated use of the same instances of axioms. Unlike Prolog, literals in goals can be marked with information that dictates how the literals are to be treated by the inference system (in Prolog, all literals in goals are treated alike and must be proved). A literal can be marked as one of the following:

proved The literal has been proved or is in the process of being proved.[2]

assumed The literal is being assumed.

unsolved The literal is neither proved nor assumed.

The initial goal clause $\leftarrow Q_1, \ldots, Q_n$ in a deduction consists of literals Q_k that are either unsolved or assumed. If any assumed literals are present, they must precede the unsolved literals. Unsolved literals must either be proved from the knowledge base, plus any assumptions that appear in the initial goal clause or are made during the proof, or, in the case of assumable literals, be directly assumed. Literals that are proved or assumed are retained in all successor goal clauses in the deduction and are used to eliminate matching

[2] In this inference system, a literal marked as proved will have been fully proved when no literal to its left remains unsolved.

goals. The final goal clause $\leftarrow P_1, \ldots, P_m$ in a deduction must consist entirely of proved or assumed literals P_k.

4.1 Inference Rules

Suppose the current goal is $\leftarrow Q_1, \ldots, Q_n$ and that Q_i is the leftmost unsolved literal. Then the following inferences are possible.

Resolution with a fact. Let Q be a fact with its variables renamed, if necessary, so that it has no variables in common with the goal $\leftarrow Q_1, \ldots, Q_n$. Then, if Q_i and Q are unifiable with most general unifier σ, the goal

$$\leftarrow Q_1\sigma, \ldots, Q_n\sigma$$

can be derived, where $Q_i\sigma$ is marked as proved.[3] The cost of the resulting goal is the cost of the original goal plus the cost of the axiom Q.

Resolution with a rule. Let $Q \leftarrow P_1, \ldots, P_m$ be a rule with its variables renamed, if necessary, so that it has no variables in common with the goal $\leftarrow Q_1, \ldots, Q_n$. Then, if Q_i and Q are unifiable with most general unifier σ, the goal

$$\leftarrow Q_1\sigma, \ldots, Q_{i-1}\sigma, P_1\sigma, \ldots, P_m\sigma, Q_i\sigma, \ldots, Q_n\sigma$$

can be derived, where $Q_i\sigma$ is marked as proved and each $P_k\sigma$ is unsolved.[4] The cost of the resulting goal is the cost of the original goal plus the cost of the axiom $Q \leftarrow P_1, \ldots, P_m$.

Making an assumption. If Q_i is assumable in the goal $\leftarrow Q_1, \ldots, Q_n$, then

$$\leftarrow Q_1, \ldots, Q_n$$

can be derived, where Q_i is assumed.[5] The cost of the resulting goal is the cost of the original goal plus the cost of assuming Q_i.

[3] Each literal Q_k or $Q_k\sigma$ in a goal resulting from one of these inference rules is proved or assumed precisely when Q_k in the parent goal is, unless it is stated otherwise.

[4] Note that the **resolution with a fact** and **resolution with a rule** operations differ from Prolog's principally in their retention of $Q_i\sigma$ (marked as proved) in the result.

[5] The same result, except for Q_i's being assumed, can be derived by the **resolution with a fact** operation if assumable literals are asserted as axioms. The final proof could be examined to distinguish between proved

Factoring with a proved or assumed literal. If Q_i and Q_j $(j < i)$[6] are unifiable with most general unifier σ, the goal

$$\leftarrow Q_1\sigma, \ldots, Q_{i-1}\sigma, Q_{i+1}\sigma, \ldots, Q_n\sigma$$

can be derived. The cost of the resulting goal is the same as the cost of the original goal. In addition, only when least specific abduction is done, Q_i can be eliminated by factoring with Q_j, where $(j > i)$ and Q_j is assumable; $Q_j\sigma$ is assumed in the result. If Q_j was already assumed in the original goal, the cost of the resulting goal is the same as the cost of the original one; otherwise it is the cost of the original goal plus the cost of assuming Q_j.

Consider again the example of explaining $Q(x) \wedge R(x) \wedge S(x)$ with R assumable from a knowledge base that includes $P(a)$, $P(x) \supset Q(x)$, and $Q(x) \wedge R(x) \supset S(x)$. Proved literals are marked by brackets [], assumed literals by braces {}.

```
1. <- Q(x), R(x), S(x).
2. <- P(x), [Q(x)], R(x), S(x).            % resolve 1 with Q(x) <- P(x)
3. <- [P(a)], [Q(a)], R(a), S(a).          % resolve 2 with P(a)
4. <- [P(a)], [Q(a)], {R(a)}, S(a).        % assume R(a) in 3
5. <- [P(a)], [Q(a)], {R(a)}, Q(a), R(a), [S(a)].
                                       % resolve 4 with S(x) <- Q(x), R(x)
6. <- [P(a)], [Q(a)], {R(a)}, R(a), [S(a)]. % factor 5
7. <- [P(a)], [Q(a)], {R(a)}, [S(a)].      % factor 6
```

The abductive proof is complete when all literals are either proved or assumed. Each axiom instance and assumption was used or made only once in the proof. The cost of the proof can be determined quickly by adding the costs of the axioms or assumed literals in each step of the proof.

If no literals are assumed, the procedure is a disguised form of Shostak's graph construction (GC) procedure [13] restricted to Horn clauses, where proved literals play the role of

and assumed literals. Although using a fact and making an assumption can be merged operationally in this way, we prefer to regard them as separate operations. An important distinction between facts and assumable literals is that facts are consistent with the [assumed-to-be-consistent] knowledge base; assumptions made in an abductive explanation should be checked for consistency with the knowledge base before being accepted.

[6] Q_j must have been proved or assumed, since it precedes Q_i.

Shostak's C-literals. It also resembles Finger's ordered residue procedure [7], except that the latter retains assumed literals (rotating them to the end of the clause) but not proved literals. Thus, it combines the GC procedure's ability to compute simple proof trees for Horn clauses with the ordered residue procedure's ability to make assumptions in abductive proofs.

5 Future Directions

Many extensions of this work are possible. The most important to us right now are a more flexible assignment of assumption costs and a procedure for dealing with non-Horn clause formulas.

5.1 Assumption Costs

The designation of which literals are assumable and the assignment of assumption costs are more rigid than we would like.

In predicate specific abduction, any literal with an assumable predicate is assumable, but its assumption cost is fixed. For example, in interpreting the sentence "The man hit another man," we would want to prove abductively a logical form such as $man(x) \wedge man(y) \wedge hit(x, y) \wedge x \neq y$. Predicate specific abduction would require that $man(x)$ and $man(y)$ be assumable with equal cost; the definite reference for the first man suggests that $man(y)$ should be assumed more easily.

In least specific abduction, only literals in the initial formula can be assumed. Although this yields correct results in many cases, it is clearly sometimes necessary to make deeper assumptions that imply the initial formula. When interpreting a piece of text, which includes references to fish and pets, with logical form

$$fish(x) \wedge pet(y) \wedge \cdots$$

we are forced to assume $fish(x)$ and $pet(y)$ if no fish or pets are in the knowledge base. But we would really like to consider the possibility that x and y refer to the same entity, i.e., a pet fish, which we could have done only if (according to our knowledge base) all fish are pets or all pets are fish, by assuming one and using it to prove the other. What is needed are axioms like

$$fish(x) \wedge fp(x) \supset pet(x) \quad \text{and} \quad pet(x) \wedge pf(x) \supset fish(x)$$

where fp and pf are predicates expressing the extra requirements for a fish to be a pet and a pet to be a fish. With the former axiom, $fish(x) \wedge pet(y) \wedge \cdots$ can be explained by assuming $fish(x)$ and $pet(y)$, as before, or by assuming $fish(x)$ and $fp(x)$, with $pet(x)$ a consequence.

Such reasoning requires that literals other than those in the original formula be assumable and that there must be a way of assigning assumption costs to them.

The method we have adopted, which has not yet been fully analyzed and is described more extensively elsewhere [9], is to allow assumability and assumption costs to be propagated from consequent literals to antecedent literals in implications.

Thus, the implication

$$P_1^{w_1} \wedge P_2^{w_2} \supset Q$$

states that P_1 and P_2 imply Q, but also that, if Q is assumable with cost c, then P_1 is assumable with cost $w_1 c$ and P_2 is assumable with cost $w_2 c$ in the result of backward chaining from Q by the implication. If $w_1 + w_2 < 1$, most specific abduction is favored, since the cost of assuming P_1 and P_2 is less than the cost of assuming Q. If $w_1 + w_2 > 1$, least specific abduction is favored: Q will be assumed in preference to P_1 and P_2. But, depending on the weights, P_i might be assumed in preference to Q if P_j is provable.

Factoring can also reduce the cost of assuming antecedent literals. When $Q \wedge R$ is explained from

$$P_1 \wedge P_2 \supset Q$$
$$P_2 \wedge P_3 \supset R$$

the cost of assuming P_1, P_2, and P_3 may be less than the cost of assuming Q and R, even though P_1 and P_2 cost more than Q, and P_2 and P_3 cost more than R.

5.2 Non-Horn Clause Proofs

Computing minimum-cost proofs from non-Horn sets of axioms is more difficult and would take us farther from Prolog-like inference systems. A mutually resolving set of clauses is a set of clauses such that each clause can be resolved with every other. Shostak [14] proved that mutually resolving sets of clauses (having no tautologies) with no single atom occurring in every clause do not have simple proof trees. This result is true of the GC procedure as well as of resolution. So, although we were able to use the GC procedure to compute simple proof trees for sets of Horn clauses, this cannot be done for non-Horn sets.

For non-Horn clause proofs, an assumption mechanism can be added to a resolution-based inference system that is complete for non-Horn clauses (such as the GC procedure or the model elimination procedure that is implemented in PTTP [15]), with more complicated rules for counting costs to compensate for the absence of simple proof trees.

Alternatively, an assumption mechanism can be added to the matings or connection method [1, 2]. These proof procedures do not require multiple occurrences of the same instances of axioms. This approach would reduce requirements on the syntactic form of the axioms (e.g., the need for clauses) so that a cost could be associated with an arbitrary axiom formula instead of a clause.

6 Conclusion

We have formulated part of the natural-language-interpretation task as abductive inference. The process of interpreting sentences in discourse can be viewed as the abductive inference of what assumptions must be made for the listener to know that the sentence is true. The forms of abduction suggested for diagnosis, and for design synthesis and planning, are generally unsuitable for natural-language interpretation. We suggest that least specific abduction, in which only literals in the logical form can be assumed, is especially useful for natural-language interpretation.

Numerical costs can be assigned to axioms and assumable literals so that the intended interpretation of a sentence will hopefully be obtained by computing the minimum-cost abductive explanation of the sentence's logical form. Axioms can be assigned different costs to reflect their relevance to the sentence. Different literals in the logical form can be assigned different assumption costs according to the form of the sentence, with literals from indefinite references being more readily assumable than those from definite references.

We presented a Prolog-like inference system that computes abductive explanations by means of either predicate specific or least specific abduction. The inference system is designed to compute the cost of an explanation correctly, so that multiple occurrences of the same instance of an axiom or assumption are not charged for more than once.

We suggested, but have not yet fully developed, an approach that extends least specific abduction to allow assumability and assumption costs to be propagated from consequent literals to antecedent literals in implications. This is intended for cases in which our preferred method of least specific abduction is unable to produce the intended interpretation.

Most of the ideas presented here have been implemented in the TACITUS project at SRI [8, 9].

Acknowledgements

This work has been greatly facilitated by discussions with Jerry Hobbs, Douglas Edwards, Todd Davies, John Lowrance, and Mabry Tyson.

References

[1] Andrews, P.B., Theorem Proving Via General Matings, *Journal of the ACM* 28:193–214 (1981).

[2] Bibel, W., *Automated Theorem Proving,* Friedr. Vieweg & Sohn, Braunschweig, West Germany, 1982.

[3] Charniak, E., Motivation Analysis, Abductive Unification, and Nonmonotonic Equality, *Artificial Intelligence* 34:275–295 (1988).

[4] Charniak, E. and Shimony, S.E. Probabilistic semantics for cost based abduction. Technical Report CS-90-02, Department of Computer Science, Brown University, Providence, Rhode Island, February 1990.

[5] Cox, P.T. and Pietrzykowski, T., Causes for Events: Their Computation and Applications, *Proceedings of the 8th Conference on Automated Deduction,* Oxford, England, July 1986, 608–621.

[6] Cox, P.T. and Pietrzykowski, T., General Diagnosis by Abductive Inference, *Proceedings of the 1987 Symposium on Logic Programming,* San Francisco, California, August 1987, 183–189.

[7] Finger, J.J., Exploiting Constraints in Design Synthesis, Ph.D. dissertation, Department of Computer Science, Stanford University, Stanford, California, February 1987.

[8] Hobbs, J.R. and Martin, P., Local Pragmatics, *Proceedings of the Tenth International Conference on Artificial Intelligence,* Milan, Italy, August 1987, 520–523.

[9] Hobbs, J.R., Stickel, M., Martin, P., and Edwards, D., Interpretation as Abduction, *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics,* Buffalo, New York, June 1988, 95–103.

[10] Maier, D. and Warren, D.S., *Computing with Logic,* Benjamin/Cummings, Menlo Park, California, 1988.

[11] Norvig, P., Inference in Text Understanding, *Proceedings of the AAAI-87 Sixth National Conference on Artificial Intelligence,* Seattle, Washington, July 1987, 561–565.

[12] Pople, H.E.,Jr., On the Mechanization of Abductive Logic, *Proceedings of the Third International Joint Conference on Artificial Intelligence,* Stanford, California, August 1973, 147–152.

[13] Shostak, R.E., Refutation Graphs, *Artificial Intelligence* 7:51–64 (1976).

[14] Shostak, R.E., On the Complexity of Resolution Derivations, unpublished.

[15] Stickel, M.E., A Prolog Technology Theorem Prover: Implementation by an Extended Prolog Compiler, *Journal of Automated Reasoning* 4:353–380 (1988).

[16] Stickel, M.E. and Tyson, W.M., An Analysis of Consecutively Bounded Depth-First Search with Applications in Automated Deduction, *Proceedings of the Ninth International Joint Conference on Artificial Intelligence,* Los Angeles, California, August 1985, 1073–1075.