

Festschrift

LNCS 5393

Jacques Calmet  
Willi Geisermann  
Jörn Müller-Quade (Eds.)

# Mathematical Methods in Computer Science

Essays in Memory of Thomas Beth



 Springer

*Commenced Publication in 1973*

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

*Lancaster University, UK*

Takeo Kanade

*Carnegie Mellon University, Pittsburgh, PA, USA*

Josef Kittler

*University of Surrey, Guildford, UK*

Jon M. Kleinberg

*Cornell University, Ithaca, NY, USA*

Alfred Kobsa

*University of California, Irvine, CA, USA*

Friedemann Mattern

*ETH Zurich, Switzerland*

John C. Mitchell

*Stanford University, CA, USA*

Moni Naor

*Weizmann Institute of Science, Rehovot, Israel*

Oscar Nierstrasz

*University of Bern, Switzerland*

C. Pandu Rangan

*Indian Institute of Technology, Madras, India*

Bernhard Steffen

*University of Dortmund, Germany*

Madhu Sudan

*Massachusetts Institute of Technology, MA, USA*

Demetri Terzopoulos

*University of California, Los Angeles, CA, USA*

Doug Tygar

*University of California, Berkeley, CA, USA*

Gerhard Weikum

*Max-Planck Institute of Computer Science, Saarbruecken, Germany*

Jacques Calmet Willi Geiselmann  
Jörn Müller-Quade (Eds.)

# Mathematical Methods in Computer Science

Essays in Memory of Thomas Beth

## Volume Editors

Jacques Calmet  
Willi Geiselman  
Jörn Müller-Quade  
Universität Karlsruhe  
Institut für Algorithmen und Kognitive Systeme  
76128 Karlsruhe, Germany  
E-mail: {calmet,geiselma,muellerq}@ira.uka.de

The illustration appearing on the cover of this book is the work of Daniel Rozenberg (DADARA).

Library of Congress Control Number: 2008940842

CR Subject Classification (1998): E.3, J.2, F.2, F.1

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743  
ISBN-10 3-540-89993-6 Springer Berlin Heidelberg New York  
ISBN-13 978-3-540-89993-8 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

springer.com

© Springer-Verlag Berlin Heidelberg 2008  
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India  
Printed on acid-free paper SPIN: 12585199 06/3180 5 4 3 2 1 0

# Preface

The conference Mathematical Methods in Computer Science (MMICS) was held in the memory of Thomas Beth during December 17–19 in Karlsruhe. The conference was meant to reflect the many interests of Thomas Beth. Even though these interests might seem diverse the mathematical methods employed and especially algebra as a language were the common denominator of all his scientific achievements. The 12 contributed talks reaching from t-designs to integrated circuits were selected from 30 submissions from 14 countries.

The contributed talks were complemented by three invited talks. Teo Mora gave a talk on “Decoding Cyclic Codes: The Cooper Philosophy” embracing the areas of coding theory and symbolic computation. These areas were especially appreciated by Thomas Beth, because they combine algebra and algorithmics. Richard Jozsa lectured about “Embedding Classical into Quantum Computation” in the area of quantum information. Quantum information was a focus of research of Tomas Beth since 1993 when he co-organized one of the earliest workshops on quantum cryptography in Dagstuhl. Quantum information became his passion in 1994 when the connection between the Fourier transformation and breaking the RSA crypto system became apparent via Shor’s algorithm, which can factor integers in polynomial time on a quantum computer. The Fourier transform and cryptography were topics that played an important role in Thomas Beth’s research and this connection, once again, justified his broad view on computer science.

We were especially delighted by the very personal talk from Fred Piper, a former colleague of Thomas Beth from the time he spent at Royal Holloway College. His talk was about “Zeros and Ones” and his abstract summarizes the scope of the conference better than we can do:

Tom was a personal friend as well as being a colleague and collaborator. He was interdisciplinary in the truest sense of the word with expertise in computer science, mathematics and physics. In this short talk I will look at those areas where our personal interests overlapped. These began with finite projective planes, generalised on to block designs and then changed (from pure mathematics) to coding theory and cryptography. The talk will be historical with little technical detail but, using zeros and ones as the theme, will try to show that the path we followed was ‘natural’.

Thomas Beth would have enjoyed this conference. His legacy should support us in our research projects and remind us to never forget the pleasure of intellectual work.

October 2008

Jacques Calmet  
Willi Geiselmann  
Jörn Müller-Quade

## In Memoriam



Prof. Dr.-Ing. habil. Dr. rer. nat. Thomas Beth, professor and long-standing spokesman of the Institut für Algorithmen und Kognitive Systeme (IAKS), was born November 16, 1949 in Hannover. He studied mathematics, physics, and medicine at the Universität Göttingen and received his Dr. rer. nat. in Mathematics from the Universität Erlangen-Nürnberg in 1978 after four years of employment as a research associate.

After receiving the degree of Dr. Ing. habil. in the area of informatics in 1984 from the same university he was appointed Professor of Computer Science at the University of London and head of the Department of Computer Science and Statistics at the Royal Holloway College, University of London. There he created the research group for cryptography.

In 1985 he took a Chair of Informatics at the Universität Karlsruhe (TH) and, together with two colleagues, co-founded the Institut für Algorithmen und Kognitive Systeme, which he has represented as a spokesman ever since.

The scientific achievements of Prof. Beth were aimed at understanding algorithmic structures in larger systems or applications. This line of research, which started with his algebraic explanation of the general Fourier transform, was continued at his institute, becoming the groundwork in modern signal and image processing. Automated tools for the decomposition of signal transforms were one result of his research that yielded efficient algorithms for different applications. New methods for medical image processing were based on these methods and the algebraic models for signal transforms. Professor Beth recognized very early the importance of the wavelet transform for data compression and pattern

classification. This research was guided by the general idea to use mathematical techniques to develop solutions for a broad spectrum of tasks in signal processing and automatically realize these in very highly integrated circuits. This homogeneous development process avoids inefficiencies and design errors to a large extent.

Cryptology was another focus in the work of Prof. Beth, where he followed an analogous approach. As in his other work he kept an eye on the applicability of his methods, which is reflected by his work in the European Institute of System Security (E.I.S.S.) that he founded in 1988 and headed since then. In his research in cryptology he successfully applied methods from the mathematical areas of combinatorics and algebra. In 1982 he organized an international cryptology conference at Castle Feuerstein, from which the renowned series of EUROCRYPT conferences emerged.

With this background Thomas Beth was early on attracted by the newly emerging field of quantum computing. This area linking informatics, mathematics and physics appealed to him, not only as a researcher, but also due to the implications quantum computing has on cryptology. Encryption mechanisms which are classically considered to be secure become insecure with respect to techniques from quantum computing.

Thomas Beth became a pioneer of quantum computing on the national level as well as internationally. His activities led to the first priority program of the Deutsche Forschungsgemeinschaft and to the first European funding program in this area. In Germany he headed the first and largest research group on quantum computing in informatics.

In the Faculty for Informatics in Karlsruhe he was one of the initiators of the new scientific field of anthropomatics. This young area uses methods and models from informatics to describe the interaction of humans with their environment to supply solutions which are well adapted for individual requirements.

Teaching and research were inseparable for Prof. Beth. Passing on his knowledge was of great concern to him and he kept up a scientific dialogue at all levels: during lectures, at his institute, in the faculty and at national and international conferences. Many of his pupils are now in high positions in science and industry.

In spite of his severe illness he was actively involved in designing the future of informatics. Unfortunately, he could pursue this task for a quarter of a century only. He died on August 17, 2005.

# Organization

MMICS 2008 was organized by the Institut für Algorithmen und Kognitive Systeme, Universität Karlsruhe, Germany.

## Organizers

Jacques Calmet	Universität Karlsruhe, Germany
Willi Geiselmann	Universität Karlsruhe, Germany
Jörn Müller-Quade	Universität Karlsruhe, Germany

## Program Committee

Jörn Müller-Quade	Universität Karlsruhe, Germany (Program Chair)
Jacques Calmet	Universität Karlsruhe, Germany
Reiner Creutzburg	Brandenburg University of Applied Sciences, Germany
Willi Geiselmann	Universität Karlsruhe, Germany
Dieter Gollmann	Technische Universität Hamburg-Harburg, Germany
María Isabel González Vasco	Universidad Rey Juan Carlos, Madrid, Spain
Markus Grassl	Österreichische Akademie der Wissenschaften, Austria
Dominik Janzing	Max Planck Institute for Biological Cybernetics, Tübingen, Germany
Dieter Jungnickel	Universität Augsburg, Germany
Andreas Klappenecker	Texas A&M University
Wolfgang Mathis	Universität Hannover, Germany
Harald Niederreiter	National University of Singapore
Markus Püschel	Carnegie Mellon University, USA
Rainer Steinwandt	Florida Atlantic University, USA
Felix Ulmer	Université Rennes 1, France
Roland Vollmar	Universität Karlsruhe, Germany

# Table of Contents

## Cryptography I

On the Security of Beth's Identification Schemes against Active and Concurrent Adversaries . . . . .	1
<i>Giovanni Di Crescenzo</i>	

## Designs

Steiner $t$ -Designs for Large $t$ . . . . .	18
<i>Michael Huber</i>	
New Spatial Configurations . . . . .	27
<i>Harald Gropp</i>	
Construction of Large Constant Dimension Codes with a Prescribed Minimum Distance . . . . .	31
<i>Axel Kohnert and Sascha Kurz</i>	

## Quantum Computing

Invited Talk: Embedding Classical into Quantum Computation . . . . .	43
<i>Richard Jozsa</i>	
A Criterion for Attaining the Welch Bounds with Applications for Mutually Unbiased Bases . . . . .	50
<i>Aleksandrs Belovs and Juris Smotrovs</i>	
An Efficient Quantum Algorithm for the Hidden Subgroup Problem over Weyl-Heisenberg Groups . . . . .	70
<i>Hari Krovi and Martin Rötteler</i>	

## Algorithms

Computing Equiangular Lines in Complex Space . . . . .	89
<i>Markus Grassl</i>	
Complexity of Comparing Monomials and Two Improvements of the Buchberger-Möller Algorithm . . . . .	105
<i>Samuel Lundqvist</i>	

## Coding Theory

Invited Talk: Decoding Cyclic Codes: The Cooper Philosophy (Extended Abstract) . . . . .	126
<i>Teo Mora and Emmanuela Orsini</i>	

Kernel Dimension for Some Families of Quaternary Reed-Muller Codes ..... 128  
*J. Pernas, J. Pujol, and M. Villanueva*

**Cryptography II**

Coding-Based Oblivious Transfer ..... 142  
*Kazukuni Kobara, Kirill Morozov, and Raphael Overbeck*

Protection of Sensitive Security Parameters in Integrated Circuits ..... 157  
*Dejan E. Lazich and Micaela Wuensche*

On Reconstruction of RC4 Keys from Internal States ..... 179  
*Shahram Khazaei and Willi Meier*

**Author Index** ..... 191

# On the Security of Beth's Identification Schemes against Active and Concurrent Adversaries

Giovanni Di Crescenzo

Telcordia Technologies, Piscataway, NJ, USA  
giovanni@research.telcordia.com

**Abstract.** One of the earliest identification schemes was proposed by Beth in [6]. Since its introduction, variations and generalizations of this scheme have been considered, and, recently, the property of security against passive impersonation was shown, under a weak unforgeability assumption on the hashed El Gamal signature scheme, for two such variants: one in the standard (i.e., not identity-based) and one in the identity-based model. However, the security of both protocols under active and concurrent impersonation attacks was left open.

In this paper we prove that very minor modifications to these schemes result in schemes that satisfy security under active and concurrent impersonation attacks, assuming a one-more-dlog assumption. The resulting protocols are just as efficient as the original variants, which are, in turn, somewhat more efficient (but less general) of the original one proposed by Beth.

## 1 Introduction

An identification scheme is a method for a party A to convince another party B of A's identity. While identification schemes are routinely used in real-life using physical proofs of identity (e.g., identity cards, driving licenses, etc.), computer technology has raised the problem of remote identification schemes; i.e., identification schemes where A and B are physically distant. As of today, several problems related to identification schemes have been studied, several security notions and schemes have been proposed, and the study of (remote) identification schemes is an important research area in Cryptography. Here, the most common scenario, which we also study in this paper, is that of A publishing a 'public key' and keeping secret a matching 'secret key', and using the secret key to identify to B.

**Security Notions.** Important problems in this area include formulating appropriate security notions for an identification scheme. A first natural notion that can be proposed is that an identification scheme is secure if no efficient adversary can impersonate A, even after witnessing many identification sessions between A and B (this notion is usually called 'security against an impersonation attack of passive type', since the adversary is passively eavesdropping sessions between A and B). Note that the adversary is not given the secret key. A second and

stronger notion is that an identification scheme is secure if no efficient adversary can impersonate A, even after taking part in many sequential identification sessions with A, playing as B (this notion is usually called ‘security against an impersonation attack of active type’, since the adversary is actively involved in the sessions between A and B before attempting the impersonation attempt). An even stronger notion is ‘security against an impersonation attack of concurrent type’, which extends the previous notion in that the identification sessions played by the adversary with A before attempting its impersonation attempt can be run concurrently with multiple entities that use A’s secret key.

**Previous work.** The first identification schemes have been given in [12,13,15] and were based on the hardness of the number-theoretic problem of quadratic residuosity modulo a composite integer. Another important contribution of [13] is the general paradigm of using zero-knowledge [15] proofs of knowledge in order to prove the knowledge of an identity without revealing its associated secret key. Since then, several improvements and variants of the mentioned schemes have been proposed, mostly motivated by efficiency considerations (we mention, in particular, [6,17,20,18], but see also references in [1]). Many of the mentioned schemes are efficient in all metrics of interest (i.e., time and communication complexity).

While some of the early identification schemes (e.g., [12,13,18]) were given a proof of security against active impersonation attacks (and can be showed to be secure against concurrent impersonation attacks), this was not immediately the case for other schemes. For instance, the popular schemes in [17,20] were only proved to be secure against active and concurrent attacks much later in [4]. Moreover, the status of other schemes, including Beth’s scheme in [6], with respect to these security notions is currently unknown. Given the special efficiency of these schemes, it remains of great interest to prove, disprove their security against advanced security notions.

**Our contribution.** We consider the identification scheme from [6], which is one of the earliest identification scheme and is based on one of the most popular signature schemes (i.e., El Gamal signatures [5]). This scheme has already received some attention in the literature, as it was revised and generalized in [7,8], and further studied in [1].

The scheme in [6] had been proposed in the identity-based model, a more complex model than the standard model discussed above, where the user’s secret key is somehow tied to the user’s identity by a trusted authority. In [1] it was observed that many identification schemes proposed in the identity-based model had a corresponding scheme in the standard model, and viceversa, and a scheme in one model could be mapped to a scheme in the other model via a particular transformation, if some specific ‘convertibility’ condition was satisfied. The authors in [1] used this approach to surface several identification schemes in one model that were related to the known scheme in the other model, proved several security results on the surfaced schemes, and left some related open problems. In particular, they proposed an identification scheme in the standard model, which we call Beth-SI-0, that is uniquely related to (a slightly more efficient and

less general variant of) the identification scheme in the identity-based model from [6]. They also proved scheme Beth-SI-0 to be secure against passive impersonation, assuming the universal unforgeability under no-message attack of the hashed-message El Gamal signature scheme [5]. Using their transformation based on the convertibility property, they also proposed an identification scheme in the identity-based model, which we call Beth-IBI-0, preserving the same type of security. Finally, they left open the security of both schemes Beth-SI-0 and Beth-IBI-0 against active and concurrent impersonation attacks.

In this paper we define a very minor, seemingly irrelevant, variation of scheme Beth-SI-0, which we call Beth-SI-1, and prove it secure in the standard model against active and concurrent impersonation attacks, under a one-more-dlog assumption (see also [4,2] for related assumptions). Here, we note that scheme Beth-SI-1 maintains the same efficiency as scheme Beth-SI-0. We then observe that scheme Beth-SI-1 also satisfies the mentioned convertibility condition and thus obtain a scheme Beth-IBI-1 for which we can prove security against active and concurrent impersonation attacks, under the same intractability assumption, in the identity-based model. Our minor modification preserves the same time and communication efficiency of the starting schemes, which are, in turn, more efficient but less general variants of Beth's original scheme [6].

**Organization of the paper.** We start with basic modeling and formal definitions in Section 2. We review the El-Gamal signature scheme [5], and the identification scheme Beth-SI-0 in Section 3. We then describe scheme Beth-SI-1 and prove its security in Section 4. Finally, we discuss the extension to scheme Beth-IBI-1 in Section 5.

## 2 Definitions

In this section we give the scenario for identification schemes, defining the entities involved, the assumed connectivity among them, the phases, the (sub)protocols, and their security requirements. We start with some basic notations.

**Basic notations.** The expression  $y \leftarrow S$  denotes the probabilistic process of randomly and independently choosing  $y$  from set  $S$ . The expression  $y \leftarrow A(x_1, x_2, \dots)$  denotes the (possibly probabilistic) process of running algorithm  $A$  on input  $x_1, x_2, \dots$  and any necessary random coins, and obtaining  $y$  as output. The expression  $z \leftarrow (A(x_1, x_2, \dots) \longleftrightarrow B(y_1, y_2, \dots))$  denotes the (possibly probabilistic) process of running an interactive protocol [15] between algorithm  $A$ , taking as input  $x_1, x_2, \dots$  and any necessary random coins, and algorithm  $B$ , taking as input  $y_1, y_2, \dots$  and any necessary random coins, where  $tr$  is the sequence of messages exchanged by  $A$  and  $B$  as a result of this execution, and  $z$  is  $B$ 's final output. If  $m_i$  denotes the  $i$ -th message sent by, say, algorithm  $A$ , in an interactive protocol  $(A(x) \longleftrightarrow B(y))$ , we also denote the process to create this message as  $m_i \leftarrow A(x, m_1, \dots, m_{i-1})$ , where  $m_1, \dots, m_{i-1}$  are the previous messages exchanged between  $A$  and  $B$ .

**System scenario and entities.** We consider an arbitrary system (or network) containing services of interest to a number of *users*. Authorization to access such services is checked by a *server*, via an execution of a 2-party protocol, called *identification scheme*, run with the interested user. Such executions can happen sequentially (each execution starting after the previous one is finished) or concurrently (the server runs at the same time one execution with each one of many users). For simplicity, we assume that the communication link between each user and the server is private or not subject to attacks, although we note that the model in which this link is also subject to adversary attacks is of orthogonal focus in the areas of cryptography and security (but is not part of the standard model for identification schemes, as studied in the cryptography area, and in this paper as well). We also denote a user with the term ‘prover’ and the server with the term ‘verifier’, since in an identification scheme the user will prove her/his identity to the server.

**Algorithms and Correctness Requirement.** Let  $\sigma$  be a security parameter, expressed in unary notation (i.e.,  $1^\sigma$ ). An *identification scheme* (with security parameter  $\sigma$ ) consists of a *setup* algorithm or subprotocol, typically run between the server and a given user, or by the user alone; and an *identification* subprotocol, the latter in turn consisting of a *prover algorithm*, run by the user/prover, and a *verifier algorithm*, run by the server/verifier.

The setup algorithms that we consider, denoted as KG, are only run by the user. On input a security parameter  $\sigma$  in unary, algorithm KG returns a public key  $pk$  and a matching secret key  $sk$ , in time at most polynomial in  $\sigma$ .

The prover algorithm P is an interactive Turing machine, as defined in [15], that, given as input  $pk, sk$ , and the messages exchanged so far with the verifier algorithm, returns a new message for the server, in time polynomial in  $\sigma$ .

The verifier algorithm V is also an interactive Turing machine, that, given as input  $pk$ , and the messages exchanged so far with the prover algorithm, returns a new message for the user running the prover algorithm. At the end of the interaction with this user, V also returns a value in  $\{\text{accept}, \text{reject}\}$ , denoting whether the server positively identifies the user or not. In both cases, V runs in time polynomial in  $\sigma$ .

Informally, the (natural) correctness requirement states that at any time, a server positively identifies users with the appropriate secret key. A formal definition follows.

**Definition 1.** Let  $\sigma$  be a security parameter and let  $IS = (KG, P, V)$  be an identification scheme. We say that IS satisfies *correctness* if it holds that

$$\text{Prob}[(pk, sk) \leftarrow KG(1^\sigma); (tr, out) \leftarrow (P(pk, sk) \longleftrightarrow V(pk)); out = \text{accept}] = 1].$$

**Security Requirements.** As typically done in the literature on identification schemes, we study security against *impersonation*; that is, against an adversary that, given all public keys (but no secret key), tries to convince the server to

be an authorized user. We consider three types of impersonation attacks<sup>1</sup>, of increasing strength:

1. *Passive Impersonation Attack*: after a pair of public and secret keys is generated, the adversary can choose to eavesdrop transcripts of executions of the identification scheme between P and V, until it decides to make an impersonation attempt; at this point, the adversary tries to make V accept without knowing the secret key.
2. *Active Impersonation Attack*: after a pair of public and secret keys is generated, the adversary can choose to engage, acting as a server, in sequential executions of the identification scheme with P until it decides to make an impersonation attempt; at this point, the adversary tries to make V accept without knowing the secret key.
3. *Concurrent Impersonation Attack*: this attack further extends the active attack in that the adversary can choose to engage, acting as a server, in concurrent executions of the identification scheme with a different instantiation of P (using the same public and private keys), until it decides to make an impersonation attempt; thus, at any given time, the adversary can decide to start a new session with a new instantiation of P, to continue a previous session by sending the next verifier's message, or to start the impersonation attempt.

Formally, for  $x \in \{ \text{passive, active, concurrent} \}$ , for any polynomial-time algorithm  $A = (A_v, A_p)$ , we define the experiment  $Exp_x^{\text{IS}, A}$ , that describes the  $x$ -type impersonation attack, and returns 1 (resp., 0) if the attack is successful (resp., not successful). We detail the experiments for  $x = \text{active}$  and  $x = \text{concurrent}$ , and then describe the minor variation needed to obtain the experiment for  $x = \text{passive}$ .

$Exp_{\text{active}}^{\text{IS}, A}(1^\sigma)$

1.  $(pk, sk) \leftarrow \text{KG}(1^\sigma)$
2.  $a \leftarrow A_v(1^\sigma, pk)$
3. while  $(a \neq \text{attack})$  do  
 $(tr, out) \leftarrow (P(pk, sk) \longleftrightarrow A_v(pk))$   
 $(a, aux) \leftarrow A_v(1^\sigma, aux, tr, out)$
4.  $(tr, out) \leftarrow (A_p(1^\sigma, aux) \longleftrightarrow V(pk))$
5. if  $out = \text{accept}$  then **return**: 1  
else **return**: 0.

$Exp_{\text{concurrent}}^{\text{IS}, A}(1^\sigma)$

1.  $(pk, sk) \leftarrow \text{KG}(1^\sigma)$
2.  $(a, j, aux) \leftarrow A_v(1^\sigma, pk)$
3.  $tr_j \leftarrow \emptyset$
4. while  $(a \neq \text{attack})$  do  
 $mes_p \leftarrow P_j(pk, sk, tr_j)$   
 $tr_j \leftarrow tr_j | mes_p$   
 $(a, mes_v, j, aux) \leftarrow A_v(1^\sigma, aux, mes_p)$   
if  $a = \text{start}$  then  $tr_j \leftarrow \emptyset$   
if  $a = \text{continue}$  then  
 $tr_j \leftarrow tr_j | mes_v$
5.  $(tr, out) \leftarrow (A_p(1^\sigma, aux) \longleftrightarrow V(pk))$
6. if  $out = \text{accept}$  then **return**: 1  
else **return**: 0.

---

<sup>1</sup> In this paper we do not consider resetting impersonation attacks. Many popular schemes based on proofs of knowledge, like the ones we consider, are immediately insecure against resetting impersonation attacks.

$Exp_{\text{passive}}^{\text{IS},A}(1^\sigma)$  is quickly obtained from  $Exp_{\text{active}}^{\text{IS},A}(1^\sigma)$  by replacing the line

$$'(tr, out) \leftarrow (P(pk, sk) \longleftrightarrow A_v(pk))'$$

in step 3 with the line

$$'(tr, out) \leftarrow (P(pk, sk) \longleftrightarrow V(pk))'.$$

We are now ready to define the security requirement for impersonation against passive, active or concurrent attacks.

**Definition 2.** Let  $\sigma$  be a security parameter and let  $\text{IS} = (\text{KG}, \text{P}, \text{V})$  be an identification scheme. For  $x \in \{\text{passive}, \text{active}, \text{concurrent}\}$ , we say that  $\text{IS}$  is *secure against an impersonation attack of type  $x$*  if for any algorithm  $A = (A_v, A_p)$ , it holds that

$$\text{Prob} \left[ b \leftarrow Exp_x^{\text{IS},A}(1^\sigma) : b = 1 \right] \leq \epsilon,$$

for some function  $\epsilon$  negligible in  $\sigma$ .

**Remarks and Performance Metrics.** An identification scheme secure against a concurrent (resp., active) attack is also secure against active (resp., passive) attack. Following previous papers in the literature, we also pay special attention to minimize the time complexity of both prover and verifier algorithms, as well as the communication complexity of the identification scheme (i.e., the length of the messages exchanged during the identification subprotocol, as a function of the security parameter).

### 3 Preliminaries

We start our analysis by recalling two preliminary schemes that will be useful to introduce our results. First, we review the El-Gamal signature scheme [5], which is used in different ways in both identification schemes described in this paper. Second, we review a recent identification scheme in the standard model, proposed in [1] and denoted as Beth-SI-0, which is obtained as a variant of Beth's identification scheme in the identity-based model [6].

#### 3.1 El-Gamal Signature Scheme

The El-Gamal signature scheme is one of the earliest and most influential digital signature schemes in the cryptography literature. We recall some notation and then describe the scheme.

**Some notation.** Let  $1^\sigma$  be a security parameter and let  $l$  be a challenge length function over the positive integers. Also, let  $G$  be a cyclic multiplicative group, and let  $q, g$  denote its order and a generator, respectively. We say that  $G$  has

prime order if  $q$  is prime. We also say that a probabilistic polynomial-time algorithm  $\text{Gen}$  is a *prime-order cyclic multiplicative group generator*, if, on input  $1^\sigma$ , generates a triple  $(\text{desc}(G), q, g)$ , where  $\text{desc}(G)$  denotes the description of a prime-order cyclic multiplicative group  $G$ ,  $q$  denotes its order and  $g$  is a generator of  $G$ .

**The El Gamal signature scheme.** We actually recall a variant of the original scheme, called *hashed-message El Gamal signature scheme* where the message is processed by hashing it into an element of  $\mathbb{Z}_q$ , using a collision-resistant hash function  $H$ . This scheme can be formally defined as a triple  $(\text{KG}, \text{Sign}, \text{Verify})$  of efficient algorithms, which are, in turn, defined as follows.

**The Algorithm KG:** On input security parameter  $1^\sigma$ , run the following instructions: let  $(\text{desc}(G), q, g) \leftarrow \text{Gen}(1^\sigma)$  and  $x \leftarrow \mathbb{Z}_q$ , set  $y = g^x$ ,  $pk = (\text{desc}(G), q, g, X)$  and  $sk = (pk, x)$ , and return:  $(pk, sk)$ .

**The Algorithm Sign:** On input  $sk = (pk, x)$  and message  $M$ , where  $pk = (\text{desc}(G), q, g)$ , run the following instructions: let  $r \leftarrow \mathbb{Z}_q$ , set  $R = g^r$  and  $s = r^{-1}(H(M) - xR) \bmod q$ , and return:  $(R, s)$

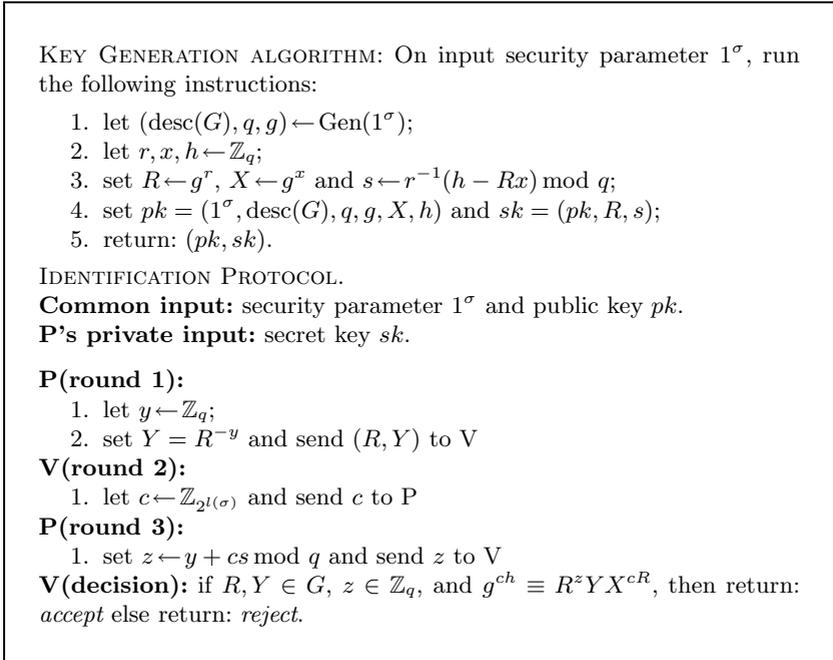
**The Algorithm Verify:** On input  $pk, M, sig$ , where  $pk = (\text{desc}(G), q, g)$ , and  $sig = (R, s)$ , check that  $X^R R^s \equiv g^{H(M)}$ . If yes, return: 1 otherwise return: 0.

The El Gamal signature scheme (without the hashing-based message preprocessing) was first presented in [5] and variants of it have been studied in several works. Yet another variant, consisting of preprocessing the message by computing  $H(R, M)$  instead of  $H(M)$ , was studied in [19] and proved to be secure, in the random oracle model, assuming the intractability of computing discrete logarithms.

### 3.2 The Identification Scheme Beth-SI-0

The identification scheme proposed in [6] is an identity-based identification scheme. In [1] an approach was proposed to uniquely bind any one in a large class of identification schemes in the standard model (also called “convertible” identification schemes) to an identification scheme in the identity-based model. By using this approach, the authors in [1] surfaced an identification scheme in the standard model that is uniquely related to (a slightly more efficient and less general variant of) the identity-based identification scheme in [6]. They also proved this scheme to be secure against passive impersonation, assuming the universal unforgeability under no-message attack of the hashed-message El Gamal signature scheme [5]. We now give some notation and then recall the formal description of the Beth-SI-0=(KG0,P0,V0) scheme.

**Description of the scheme.** In Figure 2 we formally describe the identification scheme Beth-SI-0, based on any prime-order cyclic multiplicative group generator  $\text{Gen}$  and any challenge length function  $l$ .



**Fig. 1.** The modified Beth's standard identification scheme from [1]

**Properties of the scheme.** Informally speaking, in scheme Beth-SI-0, the interaction between the prover algorithm P0 and the verifier algorithm V0 can be shown to have two properties: (1) it is a proof of knowledge of an El-Gamal signature  $(R, s)$  of the message  $h$  in the user's public key  $pk$ ; (2) it is a honest-verifier zero-knowledge proof of knowledge of the value  $s$  in the El-Gamal signature. Both properties have an important role in [1] to prove, in the random oracle model, that scheme Beth-SI-0 secure against passive impersonation assuming that the hashed-message ElGamal signature scheme is universally unforgeable under no-message attacks. However, the security of Beth-SI-0 against active and concurrent impersonation is left as an open problem in [1]. In the next section we define a minor variation of Beth-SI-0, resulting in scheme Beth-SI-1, being provable secure against active and concurrent impersonation, under appropriate assumptions.

Note that the El-Gamal signature  $(R, s)$  of  $h$  can be computed by algorithm P0 using the secret key  $sk$ . Two further aspects are worth mentioning as an introduction to the remaining schemes in the paper. First, the component  $R$  of the El-Gamal signature  $(R, s)$  is part of the secret key  $sk$ , and in different executions of scheme Beth-SI-0 the honest prover sends precisely  $R$  in the clear to the verifier (but a dishonest prover may choose to use a value  $R' \neq R$  instead). Second, the prover algorithm P0 can run multiple executions of scheme Beth-SI-0 in polynomial time when using the same El-Gamal signature  $(R, s)$  as an auxiliary input.

In the scheme Beth-SI-1 that we analyze in Section 4, we slightly modify Beth-SI-0 precisely in these two aspects. Specifically, we simply move the value  $R$  from the secret key  $sk$  to the public key  $pk$ . One positive consequence from this modification is that a dishonest prover will be easily caught if using a value  $R' \neq R$  during the identification protocol.

## 4 The Identification Scheme Beth-SI-1

In this section we present our modification to the identification scheme Beth-SI-0 in the standard model, as described in Section 3. The resulting scheme, Beth-SI-1, is proved to be secure against concurrent (and thus, active) impersonation under the one-more-dlog assumption. We obtain the following

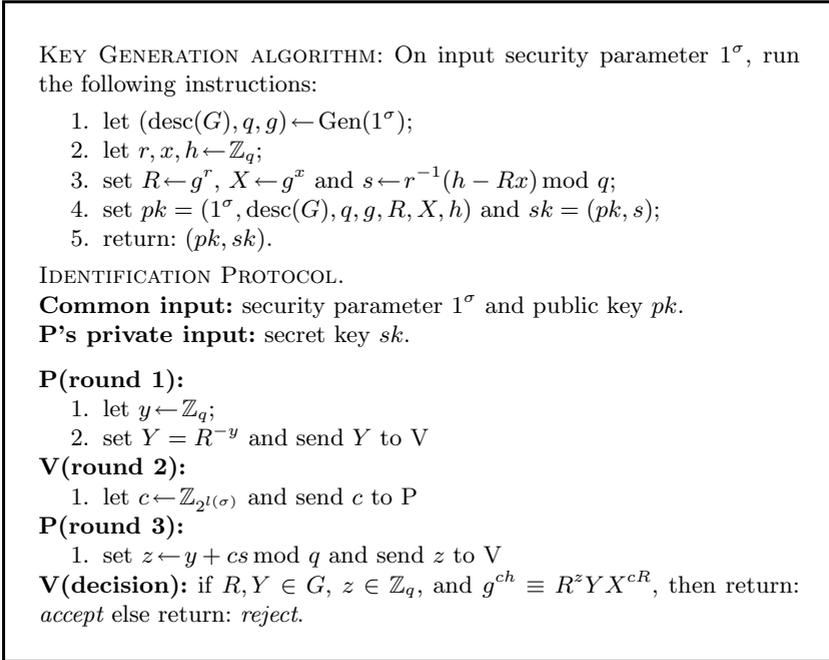
**Theorem 1.** The identification scheme IS = Beth-SI-1 with group generator Gen and challenge length function  $l$  is secure against an impersonation attack of concurrent type, under a one-more-dlog assumption. Specifically, for any adversary  $A$  running in time  $t_A$ , there exists an adversary  $B$  running in time  $t_A + (q + 1) \cdot O(k^3)$ , such that

$$\text{Prob} \left[ \text{Exp}_{\text{concurrent}}^{IS,A}(1^\sigma) = 1 \right] \leq 2^{-l(\sigma)} + \text{Prob} \left[ \text{Exp}_{\text{omd}}^B(1^\sigma) = 1 \right],$$

where  $\sigma$  denotes a security parameter,  $q$  is the number of sessions run by  $A$ ,  $k$  is the length of group elements, experiment  $\text{Exp}_{\text{concurrent}}^{IS,A}$  is defined in Section 2 and experiment  $\text{Exp}_{\text{omd}}^B$  is defined in Section 4.

In the rest of this section we prove Theorem 1. First we describe scheme Beth-SI-1, then we present a one-more-dlog assumption, and finally we prove the scheme's security under this assumption, as stated in the theorem.

**Description of scheme Beth-SI-1.** Scheme Beth-SI-1=(KG1,P1,V1) is almost identical to scheme Beth-SI-0. In particular, the interaction between the prover algorithm P1 and the verifier algorithm V1 keeps the same two above properties: (1) it is a proof of knowledge of an El-Gamal signature  $(R, s)$  of the message  $h$  in the user's public key  $pk$ ; (2) it is a honest-verifier zero-knowledge proof of knowledge of the value  $s$  in the El-Gamal signature. The main difference is as follows: in KG0, the component  $R$  of the El-Gamal signature  $(R, s)$  is part of the secret key  $sk$ , and in different executions of scheme Beth-SI-0 the honest prover sends precisely  $R$  in the clear to the verifier (but a dishonest prover may choose to use a value  $R' \neq R$ ). Instead, in KG1, the value  $R$  is part of the user's public key; thus, P1 does not need to send this value to V1 and in different executions of scheme Beth-SI-0 both the honest prover and a dishonest prover are bound to use precisely the same value  $R$ . For completeness, we still present the formal description of Beth-SI-1 in Figure 2 (here, we use the same notations as for scheme Beth-SI-0).



**Fig. 2.** The identification scheme Beth-SI-1 in the standard model

The performance properties of Beth-SI-1 are essentially the same as for Beth-SI-0. The only difference is in the communication complexity, as in Beth-SI-0, the prover sends 2 elements from group  $G$ , while in Beth-SI-1 the prover sends only 1. We now concentrate on the proof that this scheme is secure against concurrent impersonation under the one-more-dlog assumption. We start by reviewing the latter assumption.

**Our One-More-Dlog Assumption.** We use the same notations on groups as from the previous section. Informally speaking, the one-more-dlog assumption postulates the hardness of obtaining  $n$  discrete logarithms of  $n$  challenge values from group  $G$ , while the number of available queries to an oracle solving the discrete logarithm problem is strictly less than  $n$ . An assumption of this type (i.e., the one-more-RSA-inversion assumption) was first proposed in [2] and used there to prove the security of Chaum's blind signature scheme [9], and used in [4] to prove the security of the Guillou-Quisquater's digital signature scheme [17] under active and concurrent attacks. A one-more-dlog assumption was first used in [1], where it was used to prove the security of Schnorr's digital signature scheme [20] under active and concurrent attacks. Our one-more-dlog assumption considers an efficient adversary interacting with two oracles:

1. a *challenge oracle*  $C_G$  that, on input  $(1^\sigma, \text{desc}(G), q, g)$  and an empty query  $query = \perp$  from the adversary, returns a random value  $W \in G$ ,

2. an *inversion oracle*  $I_G$  that, on input  $(1^\sigma, desc(G), q, h)$ , for some generator  $h \in G$ , and a query  $query = W$  from the adversary, returns value  $w$  such that  $h^w = W$ ,<sup>2</sup>

and is formally stated as follows.

**Definition 3.** Let  $\sigma$  be a security parameter, let  $(desc(G), q, g)$  be the output of algorithm Gen on input  $1^\sigma$ , let  $C_G$  be a challenge oracle and let  $I_G$  be an inversion oracle.

Let  $Exp_{omd}^A(1^\sigma)$  denote the probabilistic experiment consisting of the following steps: first,  $(desc(G), q, g)$  is obtained as the output of algorithm Gen on input  $1^\sigma$ ; second, tuple  $(w_1, \dots, w_n)$  is obtained as the output of  $A$ , after making  $n$  queries to  $C_G$  and  $\leq n - 1$  queries to  $I_G$ , for some  $n$  polynomial in  $\sigma$ ; finally, the experiment returns 1 if  $W_i = g^{w_i}$ , for  $i = 1, \dots, n$ , where  $W_1, \dots, W_n$  were oracle  $C_G$ 's answers.

The *one-more-dlog assumption* states that for any algorithm  $A$  that has access to oracles  $C_G, I_G$  and runs in polynomial time (not counting the time needed by the oracles to answer  $A$ 's queries),

$$\text{Prob} [b \leftarrow Exp_{omd}^A(1^\sigma) : b = 1] \leq \epsilon,$$

for some function  $\epsilon$  negligible in  $\sigma$ , where the probability is over the random coins of Gen and  $A$ .

We reiterate the warnings for assumptions of similar type [2,4], by mentioning that one-more-dlog assumptions are instances of a relatively new type of assumptions. Even though the discrete logarithm problem has been studied for a long time, and is one of the few problems that cryptographers puts a significant amount of trust on, the particular type of assumption that we use requires more study before deserving the same amount of trust. Still, this assumption remains a clean and natural statement about a number-theoretic problem, which is potentially much simpler to analyze than the identification scheme based on it (a simplification that is, after all, a major goal in complexity-theoretic cryptography). Needless to say, using such an assumption to solve a problem that has been of interest since 1988 [6] and explicitly open since 2004 [1] is better than not solving the problem at all.

**Proof of Security.** We now prove that Beth-SI-1 is secure against concurrent impersonation under the one-more-dlog assumption, as claimed in Theorem 1. The proof proceeds by contradiction. We assume that Beth-SI-1 is not secure against concurrent impersonation and use the adversary breaking the Beth-SI-1 scheme to construct an adversary that violates the one-more-dlog assumption.

Formally, we assume (towards contradiction) that there exists a polynomial-time algorithm  $A = (A_v, A_p)$  for which the experiment  $Exp_x^{\text{IS}, A}$ , defined in Section 2, returns 1 with some not negligible probability. Our goal is to show there exists an algorithm  $B$  that takes  $(1^\sigma, desc(G), q, g)$  as input, makes  $n$  queries to

<sup>2</sup> Here is where our assumption differs from the assumption in [4].

**Input to B:**  $(1^\sigma, \text{desc}(G), q, g)$ , as returned by an execution of  $KG1(1^\sigma)$

**Instructions for B:**

1. make a query to oracle  $C_G$  and obtain output  $W_0$
2. let  $h, x' \leftarrow \mathbb{Z}_q$  and set  $X' \leftarrow g^{x'}$  and  $R' = W_0$
3. set  $pk = (1^\sigma, \text{desc}(G), q, R', g, X', h)$
4. repeat
  - $(a, \text{mes}, i, \text{state}) \leftarrow A_v(1^\sigma, pk, \text{state})$
  - if  $a = \text{start}$  then
    - make a query  $\perp$  to oracle  $C_G(1^\sigma, \text{desc}(G), q, g, \cdot)$  and obtain output  $W_i$
    - send  $Y = W_i$  to  $A_v$
    - $tr_i \leftarrow Y$  and  $\text{state} \leftarrow A_v(1^\sigma, tr_i, \text{state})$
  - if  $a = \text{continue}$  then
    - let  $\text{mes} = c_i \in \mathbb{Z}_{2^l(\sigma)}$ ,
    - make query  $(g^{c_i h} Y^{-1} (X')^{-c_i R'})$  to oracle  $I_G(1^\sigma, \text{desc}(G), q, R', \cdot)$
    - and obtain output  $z_i$
    - send  $z_i$  to  $A_v$
    - $tr_i \leftarrow tr_i | z_i$  and  $\text{state} \leftarrow A_v(1^\sigma, tr_i, \text{state})$
  - until  $(a = \text{attack})$
5. let  $q$  be the number of sessions run by  $A_v$
6. let  $(Y', \text{state}) \leftarrow A_p(1^\sigma, \text{state})$
7. let  $c'_1 \leftarrow \mathbb{Z}_{2^l(\sigma)}$  and send  $c'_1$  to  $A_p$
8. let  $(z'_1, \text{state}) \leftarrow A_p(1^\sigma, \text{state}, c'_1)$
9. rewind  $A_p$  to the state just after returning pair  $(Y', \text{state})$
10. let  $c'_2 \leftarrow \mathbb{Z}_{2^l(\sigma)}$  and send  $c'_2$  to  $A_p$
11. let  $(z'_2, \text{state}) \leftarrow A_p(1^\sigma, \text{state}, c'_2)$
12. if  $V1(pk, Y', c'_1, z'_1) = \text{accept}$  and  $V1(pk, Y', c'_2, z'_2) = \text{accept}$  then
  - let  $s = (z'_1 - z'_2)/(c'_1 - c'_2)$  and  $w_0 = r = (h - R'x')/s$
  - for  $i = 1, \dots, q$ ,
  - let  $w_i = c_i(h - x'R') - rz_i$
  - return:  $(w_0, w_1, \dots, w_q)$ .
  - else return:  $\perp$ .

**Fig. 3.** The algorithm  $B$  trying to break the one-more-dlog assumption

challenge oracle  $C_G$  obtaining  $W_1, \dots, W_n$  as answers, makes  $\leq n - 1$  queries to an inversion oracle  $I_G$ , runs in time polynomial in  $\sigma$  (not counting the time taken by oracles  $C_G, I_G$  to answer  $A$ 's queries), and, with probability not negligible in  $\sigma$ , returns tuple  $(w_1, \dots, w_n)$  such that  $W_i = g^{w_i}$ , for  $i = 1, \dots, n$ .

A formal description of algorithm  $B$  can be found in Figure 3. First of all,  $B$  constructs a public key  $pk$  for which the value  $X$  is replaced by a value  $X'$  for which it knows the discrete log  $x'$ , and the value  $R$  is replaced by a value  $R'$  equal to the challenge  $W_0$  returned by the oracle  $C_G$ . Then,  $A = (A_v, A_p)$ 's concurrent impersonation attack is simulated, where  $P$ 's messages to  $A_v$  are simulated by  $B$  using the challenge oracle (resp., the inversion oracle) to generate the first (resp., the second) of  $P$ 's messages in an identification session. Here, an important technical point is that the inversion oracle is queried using  $R'$  (rather than  $g$ ) as a generator. At the end of all concurrent identification sessions, algorithm  $B$

extracts from  $A_p$  a witness  $s$  and uses it, together with value  $x'$ , to compute a discrete logarithm  $r$  of  $R' = W_0$  modulo  $g$  without querying the inversion oracle. This value  $r$  is then used, together with value  $x'$ , to compute discrete logarithms of  $W_1, \dots, W_q$  modulo  $g$ . This implies that  $B$  obtains  $q + 1$  discrete logarithms, even if making  $q + 1$  queries to the challenge oracle and only  $q$  queries to the inversion oracle, thus turning  $A$ 's success in its concurrent attack into a violation of the one-more-dlog assumption.

Let  $IS$  denote the scheme Beth-SI-1. Recall that we assume (towards contradiction) that adversary  $A = (A_v, A_p)$  is such that  $\text{Prob} \left[ \text{Exp}_{\text{concurrent}}^{IS,A}(1^\sigma) = 1 \right]$  is not negligible in  $\sigma$ . We now would like to prove that  $\text{Prob} \left[ \text{Exp}_{\text{omd}}^B(1^\sigma) = 1 \right]$  is not negligible in  $\sigma$ .

*Notations.* We formally define two probabilities: the probability  $\text{Prob}[acc] = \text{Prob} \left[ \text{AccExp}^{IS}(1^\sigma) = 1 \right]$  that V1 accepts in an execution of protocol Beth-SI-1, when P1 and V1 are given additional state-related inputs  $st_p, st_v$ , respectively; and the probability  $\text{Prob}[res] = \text{Prob} \left[ \text{ResExp}^{IS}(1^\sigma) = 1 \right]$  that V1 accepts in both executions of protocol Beth-SI-1, when P1 and V1 are given additional state-related inputs  $st_p, st_v$ , respectively, and P1 is rewinded by V1. Experiments  $\text{AccExp}^{IS}(1^\sigma), \text{ResExp}^{IS}(1^\sigma)$  are defined as follows.

$\text{AccExp}^{IS}(1^\sigma)$

1.  $Y \leftarrow P1(pk, st_p)$
2.  $c \leftarrow V1(pk, st_v, Y)$
3.  $z \leftarrow P1(pk, st_p, Y, c)$
4. if  $V1(pk, st_v, Y, c, z) = \text{accept}$  then  
**return:** 1  
else **return:** 0.

$\text{ResExp}^{IS}(1^\sigma)$

1.  $Y \leftarrow P1(pk, st_p)$
2.  $c_1 \leftarrow V1(pk, st_v, Y)$
3.  $z_1 \leftarrow P1(pk, st_p, Y, c_1)$
2.  $c_2 \leftarrow V1(pk, st_v, Y)$
3.  $z_2 \leftarrow P1(pk, st_p, Y, c_2)$
4. if  $V1(pk, st_v, Y, c_1, z_1) = \text{accept}$  then  
if  $V1(pk, st_v, Y, c_2, z_2) = \text{accept}$  then  
**return:** 1
5. **return:** 0.

*Analysis.* First of all, we show that algorithm  $B$  perfectly simulates the view of algorithm  $A$  during experiment  $\text{Exp}_{\text{concurrent}}^{IS,A}(1^\sigma)$ . This is easily seen for the messages given from  $B$  to  $A_p$  as these are computed in exactly the same way in both experiments. To complete the proof of this fact, we consider the messages sent from  $B$  to  $A_v$ . There are two types of such messages: message  $Y$ , corresponding to the first prover's message in protocol Beth-SI-1, and message  $z$ , corresponding to the second prover's message in protocol Beth-SI-1. Note that in experiment  $\text{Exp}_{\text{concurrent}}^{IS,A}$ , message  $Y$  is computed as equal to  $R^{-y}$ , for some random  $y \in \mathbb{Z}_q$ , and is thus uniformly distributed in group  $G$ , which is the same distribution as the value  $Y = W_0$  returned by oracle  $C_G$  in experiment  $\text{Exp}_{\text{omd}}^B$ . Furthermore, note that in experiment  $\text{Exp}_{\text{concurrent}}^{IS,A}$ , message  $z$  is computed as the value such that  $g^{ch} \equiv R^z Y X^{cR}$ , which is the discrete logarithm (modulo  $R$ ) of  $g^{ch} Y^{-1} (X)^{-cR}$ . This latter value is precisely how a query to oracle  $I_G$  is computed by algorithm  $B$ , resulting in the answer  $z_i$  given to  $A_v$  in experiment  $\text{Exp}_{\text{omd}}^B$ .

This analysis implies the following two facts.

**Fact 1.**  $\text{Prob} \left[ \text{Exp}_{\text{concurrent}}^{IS,A}(1^\sigma) = 1 \right] = \text{Prob}[\text{acc}]$ .

**Fact 2.**  $\text{Prob} \left[ \text{Exp}_{\text{omd}}^B(1^\sigma) = 1 \right] = \text{Prob}[\text{res}]$ .

We now need a result relating the probability  $\text{Prob}[\text{acc}]$  that  $A_p$  makes verifier V1 accept to the probability  $\text{Prob}[\text{res}]$  that  $B$  can obtain two accepting conversations when rewinding  $A_p$ . This is obtained as an application of the Reset Lemma from [4] (see also [3,19]) that applies to all 3-message public-coin protocols, including Beth-SI-1. As a corollary of this lemma, we have the following

**Fact 3.** [4]  $\text{Prob}[\text{acc}] \leq 2^{-l(\sigma)} + \text{Prob}[\text{res}]$ .

By combining Facts 1, 2, 3, we obtain that

$$\text{Prob} \left[ \text{Exp}_{\text{concurrent}}^{IS,A}(1^\sigma) = 1 \right] \leq 2^{-l(\sigma)} + \text{Prob} \left[ \text{Exp}_{\text{omd}}^B(1^\sigma) = 1 \right],$$

as in the statement of Theorem 1. (Note that if  $\text{Prob} \left[ \text{Exp}_{\text{concurrent}}^{IS,A}(1^\sigma) = 1 \right]$  is not negligible and  $l(\sigma) = \omega(\log \sigma)$ , then  $\text{Prob} \left[ \text{Exp}_{\text{omd}}^B(1^\sigma) = 1 \right]$  is also not negligible.)

To complete the proof, we only need to calculate  $B$ 's running time. We note that  $B$  runs  $A_v, A_p$ , makes queries to oracles  $C_G, I_G$  (recall that we do not count the running time needed to answer such queries), and performs additions, multiplications modulo  $q$  and multiplications and exponentiations in the group  $G$ , the latter being those that asymptotically dominate the running time. Overall,  $B$  performs a constant number of exponentiations per session (see step 4) and 1 additional exponentiation (see step 2), so  $B$ 's running time can be bounded by  $t_A + (q + 1) \cdot O(k^3)$ , where  $k$  is the length of an element in group  $G$ .

## 5 The Identification Scheme Beth-IBI-1

In this section we sketch a corollary of Theorem 1. Specifically, we consider a natural transformation of the scheme Beth-SI-1 in the standard model into a scheme, denoted as Beth-IBI-1, in the identity-based model. This transformation, which we call the *standard-to-identity-based conversion*, has been used by [1] (and earlier, in the context of signatures, by [11]) and has the following attractive *security preservation property*: if the original identification scheme is secure in the standard model against an impersonation attack of passive (resp., active) (resp., concurrent) type, then the resulting identification scheme is secure in the identity-based model against an impersonation attack of passive (resp., active) (resp., concurrent) type. The transformation only applies to a class of schemes that have a special *convertibility* property, as defined in [1], and its security analysis only applies in the random oracle model.

**Sketch of technical details.** Our starting point is scheme Beth-SI-1, which is a (more efficient but less general) variant in the standard model of the original

identification scheme from [6]. We then note that this scheme enjoys the mentioned convertibility property, and invoke the standard-to-identity-based conversion to obtain a scheme Beth-IBI-1. A similar approach was already used in [1] to generate a scheme Beth-IBI-0, starting from the scheme Beth-SI-0 (also recalled in Section 3) for which the authors proved security against passive impersonation, assuming the unforgeability of the ElGamal signature scheme against no-message attacks, and using the above security preservation property. Analogously, we combine this latter property with Theorem 1 to prove that scheme Beth-IBI-1 is secure against impersonation attacks of concurrent (and thus, active) type, under the one-more-dlog assumption. The only new fact to establish to obtain this result is proving that scheme Beth-SI-1 enjoys the convertibility property. This does not logically follow from the fact that Beth-SI-0 enjoys the convertibility property, but the technique used to prove this latter claim requires only a few natural modifications to work for Beth-SI-1 as well (details omitted here). In what follows, we formally describe scheme Beth-IBI-1, which

**SKG ALGORITHM:** On input security parameter  $1^\sigma$ , run the following instructions:

1. let  $(\text{desc}(G), q, g) \leftarrow \text{Gen}(1^\sigma)$ ;
2. let  $r, x \leftarrow \mathbb{Z}_q$ ;
3. set  $R \leftarrow g^r$  and  $X \leftarrow g^x$ ;
4. set  $\text{mpk} = (1^\sigma, \text{desc}(G), q, g, R, X)$  and  $\text{msk} = (pk, r, x)$ ;
5. return:  $(\text{mpk}, \text{msk})$ .

**UKG ALGORITHM:** On input security parameter  $1^\sigma$ , random oracle  $H$ , and  $\text{mpk}, \text{msk}, id$ , run the following instructions:

1. set  $h \leftarrow H(id)$  and  $s \leftarrow r^{-1}(h - Rx) \bmod q$ ;
2. set  $\text{usk} = (\text{mpk}, id, h, s)$ ;
3. return:  $\text{usk}$ .

**IDENTIFICATION PROTOCOL.**

**Common input:** security parameter  $1^\sigma$ , master public key  $\text{mpk}$  and identity  $id$ .

**P's private input:** user secret key  $\text{usk}$ .

**P(round 1):**

1. let  $y \leftarrow \mathbb{Z}_q$ ;
2. set  $Y = R^{-y}$  and send  $Y$  to V

**V(round 2):**

1. let  $c \leftarrow \mathbb{Z}_{2^{\ell(\sigma)}}$  and send  $c$  to P

**P(round 3):**

1. set  $z \leftarrow y + cs \bmod q$  and send  $z$  to V

**V(decision):** compute  $h = H(id)$ ; if  $R, Y \in G$ ,  $z \in \mathbb{Z}_q$ , and  $g^{ch} \equiv R^z Y X^{cR}$ , then return: *accept* else return: *reject*.

**Fig. 4.** The identification scheme Beth-IBI-1 in the identity-based model

is almost identical to scheme Beth-IBI-0, just like the corresponding schemes in the standard model.

**Formal description of scheme Beth-IBI-1.** We first briefly recall the algorithms in an identification scheme in the identity-based model. Here, an identification scheme is a 4-tuple (rather than triple), consisting of two setup algorithms, one prover and one verifier algorithm.

The server setup algorithm, denoted as  $\text{sKG}$ , is only run by the server. On input a security parameter  $\sigma$  in unary, algorithm  $\text{sKG}$  returns a master public key  $mpk$  and a master secret key  $msk$  in time at most polynomial in  $\sigma$ .

The user setup algorithm, denoted as  $\text{UKG}$ , is also run by the server. On input a security parameter  $\sigma$  in unary, public key  $mpk$ , secret key  $msk$ , and a user identity  $id$ , algorithm  $\text{UKG}$  returns a user secret key  $sk_{id}$  in time at most polynomial in  $\sigma$ .

The prover and verifier algorithms  $P, V$  are as in the standard model, with the only addition that they both take identity  $id$  as an additional input.

We formally describe scheme Beth-IBI-1 in Figure 4.

**Acknowledgements.** Many thanks go to the MMICS 08 anonymous referees for their useful comments on the submitted version of this paper.

## References

1. Bellare, M., Namprempre, C., Neven, G.: Security proofs for identity-based identification and signature schemes. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 268–286. Springer, Heidelberg (2004)
2. Bellare, M., Namprempre, C., Pointcheval, D., Semanko, M.: The One-More-RSA-Inversion Problems and the Security of Chaum’s Blind Signature Scheme. *Journal of Cryptology* 16(3), 185–215 (2003)
3. Bellare, M., Miner, S.: A forward-secure digital signature scheme. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, p. 431. Springer, Heidelberg (1999)
4. Bellare, M., Palacio, A.: GQ and schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, p. 162. Springer, Heidelberg (2002)
5. El Gamal, T.: A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory* 31, 469–472 (1985)
6. Beth, T.: Efficient zero-knowledged identification scheme for smart cards. In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 77–84. Springer, Heidelberg (1988)
7. Burmester, M., Desmedt, Y., Beth, T.: Efficient Zero-Knowledge Identification Schemes for Smart Cards. *The Computer Journal* 35(1), 21–29 (1992)
8. Burmester, M., Desmedt, Y., Piper, F., Walker, M.: A General Zero-Knowledge Schemes. *Designs, Codes and Cryptography* 12(1) (1997)
9. Chaum, D.: Blind Signatures for Untraceable Payments. In: Proc. of CRYPTO 1982, Plemum, NY (1983)
10. De Santis, A., Di Crescenzo, G., Persiano, G.: Communication-Efficient Anonymous Group Identification. In: Proc. of 1998 ACM Conference on Computer and Communications Security. ACM Press, New York (1998)

11. Dodis, Y., Katz, J., Xu, S., Yung, M.: Strong key-insulated signature schemes. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 130–144. Springer, Heidelberg (2002)
12. Feige, U., Shamir, A.: Witness Indistinguishable and Witness Hiding Protocols. In: Proc. of the 22nd Annual ACM Symposium on the Theory of Computing (STOC 1990) (1990)
13. Feige, U., Fiat, A., Shamir, A.: Zero Knowledge Proofs of Identity. *Journal of Cryptology* 1(2), 77–94 (1988)
14. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
15. Goldwasser, S., Micali, S., Rackoff, C.: The Knowledge Complexity of Interactive Proof-Systems. *SIAM J. on Computing* 18(1) (1989)
16. Goldwasser, S., Micali, S., Rivest, R.: A Digital Signature Scheme Secure against Adaptive Chosen-Message Attacks. *SIAM Journal of Computing* 17(2) (1988)
17. Guillou, L.C., Quisquater, J.: A Paradoxical Identity-Based Signature Scheme Resulting from Zero-Knowledge. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 216–231. Springer, Heidelberg (1990)
18. Okamoto, T.: Provably secure and practical identification schemes and corresponding signature schemes. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 31–53. Springer, Heidelberg (1993)
19. Pointcheval, D., Stern, J.: Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology* 13(3), 361–396 (2000)
20. Schnorr, C.: Efficient Signature Generation by Smart Cards. *Journal of Cryptology* 4(3), 161–174 (1991)
21. Schnorr, C.: Security of  $2^t$ -root identification and signatures. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 143–157. Springer, Heidelberg (1996)
22. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)

# Steiner $t$ -Designs for Large $t$

Michael Huber\*

Institut für Mathematik, MA6-2, Technische Universität Berlin,  
Straße des 17. Juni 136, D-10623 Berlin, Germany  
mhuber@math.tu-berlin.de

**Abstract.** One of the most central and long-standing open questions in combinatorial design theory concerns the existence of Steiner  $t$ -designs for large values of  $t$ . Although in his classical 1987 paper, L. Teirlinck has shown that non-trivial  $t$ -designs exist for all values of  $t$ , no non-trivial Steiner  $t$ -design with  $t > 5$  has been constructed until now. Understandingly, the case  $t = 6$  has received considerable attention. There has been recent progress concerning the existence of highly symmetric Steiner 6-designs: It is shown in [M. Huber, *J. Algebr. Comb.* **26** (2007), pp. 453–476] that no non-trivial flag-transitive Steiner 6-design can exist. In this paper, we announce that essentially also no block-transitive Steiner 6-design can exist.

## 1 Introduction

One of the most central and long-standing open questions in combinatorial design theory concerns the existence of Steiner  $t$ -designs for large values of  $t$ . Although in his classical 1987 paper, L. Teirlinck [46] has shown that non-trivial  $t$ -designs exist for all values of  $t$ , no non-trivial Steiner  $t$ -design with  $t > 5$  has been constructed until now. Understandingly, the case  $t = 6$  has received considerable attention. There has been recent progress concerning the existence of highly symmetric Steiner 6-designs: The author [25] showed that no non-trivial flag-transitive Steiner 6-design can exist. Moreover, he classified all flag-transitive Steiner  $t$ -designs with  $t > 2$  (see [22,23,24,25,26] and [28] for a monograph). These results answer a series of 40-year-old problems and generalize theorems of J. Tits [47] and H. Lüneburg [39]. Earlier, F. Buekenhout, A. Delandtsheer, J. Doyen, P. Kleidman, M. Liebeck, and J. Saxl [6,15,35,37,43] had essentially characterized all flag-transitive Steiner 2-designs. All these classification results rely on the classification of the finite simple groups.

In this paper, we announce that essentially no block-transitive Steiner 6-design can exist. This confirms a far-reaching conjecture of P. Cameron and C. Praeger [10], stating that there are no non-trivial block-transitive 6-designs, for the important case of Steiner designs. Consequently, a further significant step towards

---

\* The author gratefully acknowledges support by the Deutsche Forschungsgemeinschaft (DFG).

an answer to the fundamental open question “Does there exist any non-trivial Steiner 6-design?” is provided – at least in the case of highly symmetric designs, quoting arguably Gian-Carlo Rota [42]:

“A combinatorial object without symmetries doesn’t exist - by definition.”

## 2 Combinatorial Designs

The study of combinatorial designs deals with a crucial problem of combinatorial theory, that of arranging objects into patterns according to specified rules. This is a subject of considerable interest in discrete mathematics and computer science, amongst others. In particular, there are close connections of design theory with graph theory [11,48], finite and incidence geometry [5,16], group theory [9,12,17,50], coding theory [11,27,29,30], information theory [14], cryptography [40,45], as well as classification algorithms [34].

Combinatorial designs may be regarded as generalizations of finite projective planes. More formally: For positive integers  $t \leq k \leq v$  and  $\lambda$ , we define a  $t$ - $(v, k, \lambda)$  design to be a finite incidence structure  $\mathcal{D} = (X, \mathcal{B}, I)$ , where  $X$  denotes a set of *points*,  $|X| = v$ , and  $\mathcal{B}$  a set of *blocks*,  $|\mathcal{B}| = b$ , with the following regularity properties: each block  $B \in \mathcal{B}$  is incident with  $k$  points, and each  $t$ -subset of  $X$  is incident with  $\lambda$  blocks. A *flag* of  $\mathcal{D}$  is an incident point-block pair  $(x, B) \in I$  with  $x \in X$  and  $B \in \mathcal{B}$ .

For historical reasons, a  $t$ - $(v, k, \lambda)$  design with  $\lambda = 1$  is called a *Steiner  $t$ -design* (sometimes also a *Steiner system*). We note that in this case each block is determined by the set of points which are incident with it, and thus can be identified with a  $k$ -subset of  $X$  in a unique way. If  $t < k < v$ , then we speak of a *non-trivial Steiner  $t$ -design*. As a simple example, the vector space  $Z_2^n$  ( $n \geq 3$ ) with block set  $\mathcal{B}$  taken to be the set of all subsets of four distinct elements of  $Z_2^n$  whose vector sum is zero is a (boolean) Steiner 3- $(2^n, 4, 1)$  design. There are many infinite classes of Steiner  $t$ -designs for  $t = 2$  and 3, however for  $t = 4$  and 5 only a finite number are known. For a detailed treatment of combinatorial designs, we refer to [1,13,20,31,44]. In particular, [1,13] provide encyclopedic accounts of key results and contain existence tables with known parameter sets.

In what follows, we are interested in  $t$ -designs which admit groups of automorphisms with sufficiently strong symmetry properties such as transitivity on the blocks or on the flags. We consider automorphisms of a  $t$ -design  $\mathcal{D}$  as pairs of permutations on  $X$  and  $\mathcal{B}$  which preserve incidence, and call a group  $G \leq \text{Aut}(\mathcal{D})$  of automorphisms of  $\mathcal{D}$  *block-transitive* (respectively *flag-transitive*, *point  $t$ -transitive*, *point  $t$ -homogeneous*) if  $G$  acts transitively on the blocks (respectively transitively on the flags,  $t$ -transitively on the points,  $t$ -homogeneously on the points) of  $\mathcal{D}$ . For short,  $\mathcal{D}$  is said to be, e.g., block-transitive if  $\mathcal{D}$  admits a block-transitive group of automorphisms.

### 3 Basic Properties and Existence Results

We give some basic properties and known results concerning the existence of  $t$ -designs which are important for the remainder of the paper.

If  $\mathcal{D} = (X, \mathcal{B}, I)$  is a  $t$ - $(v, k, \lambda)$  design with  $t \geq 2$ , and  $x \in X$  arbitrary, then the *derived* design with respect to  $x$  is  $\mathcal{D}_x = (X_x, \mathcal{B}_x, I_x)$ , where  $X_x = X \setminus \{x\}$ ,  $\mathcal{B}_x = \{B \in \mathcal{B} : (x, B) \in I\}$  and  $I_x = I|_{X_x \times \mathcal{B}_x}$ . In this case,  $\mathcal{D}$  is also called an *extension* of  $\mathcal{D}_x$ . Obviously,  $\mathcal{D}_x$  is a  $(t-1)$ - $(v-1, k-1, \lambda)$  design.

For  $\mathcal{D} = (X, \mathcal{B}, I)$  a Steiner  $t$ -design with  $G \leq \text{Aut}(\mathcal{D})$ , let  $G_x$  denote the stabilizer of a point  $x \in X$ , and  $G_B$  the setwise stabilizer of a block  $B \in \mathcal{B}$ . For  $x, y \in X$  and  $B \in \mathcal{B}$ , we define  $G_{xy} = G_x \cap G_y$  and  $G_{xB} = G_x \cap G_B$ .

For any  $x \in \mathbb{R}$ , let  $\lfloor x \rfloor$  denote the greatest positive integer which is at most  $x$ .

All other notations remain as defined in Sect. 2.

Basic necessary conditions for the existence of  $t$ -designs can be obtained via elementary counting arguments (see, for instance, [1]):

**Proposition 1.** *Let  $\mathcal{D} = (X, \mathcal{B}, I)$  be a  $t$ - $(v, k, \lambda)$  design, and for a positive integer  $s \leq t$ , let  $S \subseteq X$  with  $|S| = s$ . Then the total number of blocks incident with each element of  $S$  is given by*

$$\lambda_s = \lambda \frac{\binom{v-s}{t-s}}{\binom{k-s}{t-s}}.$$

In particular, for  $t \geq 2$ , a  $t$ - $(v, k, \lambda)$  design is also an  $s$ - $(v, k, \lambda_s)$  design.

It is customary to set  $r := \lambda_1$  denoting the total number of blocks incident with a given point (referring to the ‘replication number’ from statistical design of experiments, one of the origins of design theory).

**Corollary 1.** *Let  $\mathcal{D} = (X, \mathcal{B}, I)$  be a  $t$ - $(v, k, \lambda)$  design. Then the following holds:*

(a)  $bk = vr$ .

(b)  $\binom{v}{t} \lambda = b \binom{k}{t}$ .

(c)  $r(k-1) = \lambda_2(v-1)$  for  $t \geq 2$ .

**Corollary 2.** *Let  $\mathcal{D} = (X, \mathcal{B}, I)$  be a  $t$ - $(v, k, \lambda)$  design. Then*

$$\lambda \binom{v-s}{t-s} \equiv 0 \pmod{\binom{k-s}{t-s}}$$

for each positive integer  $s \leq t$ .

For non-trivial Steiner  $t$ -designs lower bounds for  $v$  in terms of  $k$  and  $t$  can be given (see P. Cameron [7, Thm. 3A.4], and J. Tits [47, Prop. 2.2]):

**Theorem 1.** *If  $\mathcal{D} = (X, \mathcal{B}, I)$  is a non-trivial Steiner  $t$ -design, then the following holds:*

(a) (Tits 1964):  $v \geq (t + 1)(k - t + 1)$ .

(b) (Cameron 1976):  $v - t + 1 \geq (k - t + 2)(k - t + 1)$  for  $t > 2$ . *If equality holds, then  $(t, k, v) = (3, 4, 8), (3, 6, 22), (3, 12, 112), (4, 7, 23),$  or  $(5, 8, 24)$ .*

We note that (a) is stronger for  $k < 2(t - 1)$ , while (b) is stronger for  $k > 2(t - 1)$ . For  $k = 2(t - 1)$  both assert that  $v \geq t^2 - 1$ .

The following result by R. Fisher [18] is classical, generally known as ‘‘Fisher’s Inequality’’:

**Theorem 2.** (Fisher 1940). *If  $\mathcal{D} = (X, \mathcal{B}, I)$  is a non-trivial  $2$ - $(v, k, \lambda)$  design, then  $b \geq v$ , that is, there are at least as many blocks as points in  $\mathcal{D}$ .*

An important generalization to arbitrary  $t$ -designs is due to D. Ray-Chaudhuri and R. Wilson [41, Thm. 1]:

**Theorem 3.** (Ray-Chaudhuri & Wilson 1975). *Let  $\mathcal{D} = (X, \mathcal{B}, I)$  be a  $t$ - $(v, k, \lambda)$  design. If  $t$  is even, say  $t = 2s$ , and  $v \geq k + s$ , then  $b \geq \binom{v}{s}$ . If  $t$  is odd, say  $t = 2s + 1$ , and  $v - 1 \geq k + s$ , then  $b \geq 2\binom{v-1}{s}$ .*

Exploration of the construction of  $t$ -designs for large values of  $t$  led to L. Teirlinck’s celebrated theorem [46], one of the major results in design theory:

**Theorem 4.** (Teirlinck 1987). *For every positive integer value of  $t$ , there exists a non-trivial  $t$ -design.*

However, although Teirlinck’s recursive methods are constructive, they only produce examples with tremendously large values of  $\lambda$ . Until now no non-trivial Steiner  $t$ -design with  $t > 5$  has been constructed.

**Research Problem.** *Does there exist any non-trivial Steiner 6-design?*

## 4 Approach via Symmetry

Besides recursive and set-theoretical approaches, many existence results for  $t$ -designs with large  $t$  have been obtained in recent years by the method of orbiting under a group (see, e.g., [2], [13, II.4]). Specifically, the consideration of  $t$ -designs which admit groups of automorphisms with sufficiently strong symmetry properties seems to be of great importance in our context - quoting arguably Gian-Carlo Rota [42]:

‘‘A combinatorial object without symmetries doesn’t exist - by definition.’’

We first state (cf. [49]):

**Proposition 2.** *Let  $t$  be a positive integer, and  $G$  a finite (abstract) group. Then there is a  $t$ -design such that the full group  $\text{Aut}(\mathcal{D})$  of automorphisms has a subgroup isomorphic to  $G$ .*

One of the early important results regarding highly symmetric designs is due to R. Block [3, Thm. 2]:

**Proposition 3.** (Block 1965). *Let  $\mathcal{D} = (X, \mathcal{B}, I)$  be a non-trivial  $t$ - $(v, k, \lambda)$  design with  $t \geq 2$ . If  $G \leq \text{Aut}(\mathcal{D})$  acts block-transitively on  $\mathcal{D}$ , then  $G$  acts point-transitively on  $\mathcal{D}$ .*

For a  $2$ - $(v, k, 1)$  design  $\mathcal{D}$ , it is elementary that the point  $2$ -transitivity of  $G \leq \text{Aut}(\mathcal{D})$  implies its flag-transitivity. For  $2$ - $(v, k, \lambda)$  designs, this implication remains true if  $r$  and  $\lambda$  are relatively prime (see, e.g., [16, Chap. 2.3, Lemma 8]). However, for  $t$ - $(v, k, \lambda)$  designs with  $t \geq 3$ , it can be deduced from Proposition 3 that always the converse holds (see [4] or [22, Lemma 2]):

**Proposition 4.** *Let  $\mathcal{D} = (X, \mathcal{B}, I)$  be a non-trivial  $t$ - $(v, k, \lambda)$  design with  $t \geq 3$ . If  $G \leq \text{Aut}(\mathcal{D})$  acts flag-transitively on  $\mathcal{D}$ , then  $G$  acts point  $2$ -transitively on  $\mathcal{D}$ .*

Investigating highly symmetric  $t$ -designs for large values of  $t$ , P. Cameron and C. Praeger [10, Thm. 2.1] deduced from Theorem 3 and Proposition 3 the following assertion:

**Proposition 5.** (Cameron & Praeger 1993). *Let  $\mathcal{D} = (X, \mathcal{B}, I)$  be a  $t$ - $(v, k, \lambda)$  design with  $t \geq 2$ . Then, the following holds:*

- (a) *If  $G \leq \text{Aut}(\mathcal{D})$  acts block-transitively on  $\mathcal{D}$ , then  $G$  also acts point  $\lfloor t/2 \rfloor$ -homogeneously on  $\mathcal{D}$ .*
- (b) *If  $G \leq \text{Aut}(\mathcal{D})$  acts flag-transitively on  $\mathcal{D}$ , then  $G$  also acts point  $\lfloor (t+1)/2 \rfloor$ -homogeneously on  $\mathcal{D}$ .*

As for  $t \geq 7$  the flag-transitivity, respectively for  $t \geq 8$  the block-transitivity of  $G \leq \text{Aut}(\mathcal{D})$  implies at least its point 4-homogeneity, they obtained the following restrictions as a fairly direct consequence of the finite simple group classification (cf. [10, Thm. 1.1]):

**Theorem 5.** (Cameron & Praeger 1993). *Let  $\mathcal{D} = (X, \mathcal{B}, I)$  be a  $t$ - $(v, k, \lambda)$  design. If  $G \leq \text{Aut}(\mathcal{D})$  acts block-transitively on  $\mathcal{D}$  then  $t \leq 7$ , while if  $G \leq \text{Aut}(\mathcal{D})$  acts flag-transitively on  $\mathcal{D}$  then  $t \leq 6$ .*

Moreover, they formulated the following far-reaching conjecture (cf. [10, Conj. 1.2]):

**Conjecture.** (Cameron & Praeger 1993). *There are no non-trivial block-transitive 6-designs.*

It has been shown recently by the author [25] that no non-trivial flag-transitive Steiner 6-design can exist. Moreover, he classified all flag-transitive Steiner

$t$ -designs with  $t > 2$  (see [22,23,24,25,26] and [28] for a monograph). These results make use of the classification of all finite 2-transitive permutation groups, which itself relies on the finite simple group classification. The characterizations answer a series of 40-year-old problems and generalize theorems of J. Tits [47] and H. Lüneburg [39]. Earlier, F. Buekenhout, A. Delandtsheer, J. Doyen, P. Kleidman, M. Liebeck, and J. Saxl [6,15,35,37,43] had essentially characterized all finite flag-transitive linear spaces, that is flag-transitive Steiner 2-designs. Their result, which also relies on the finite simple group classification, starts with a classical result of Higman and McLaughlin [21] and uses the O’Nan-Scott Theorem for finite primitive permutation groups. For the incomplete case with a 1-dimensional affine group of automorphisms, we refer to [6, Sect. 4] and [33, Sect. 3].

## 5 Non-existence of Block-Transitive Steiner 6-Designs

We assert the following main result:

**Main Theorem.** *Let  $\mathcal{D} = (X, \mathcal{B}, I)$  be a non-trivial Steiner 6-design. Then  $G \leq \text{Aut}(\mathcal{D})$  cannot act block-transitively on  $\mathcal{D}$ , except possibly when  $G = P\Gamma L(2, p^e)$  with  $p = 2$  or  $3$  and  $e$  is an odd prime power.*

We will briefly outline the main ingredients of the proof. The long and technical details will appear elsewhere:

- In order to investigate block-transitive Steiner 6-designs, we can in view of Proposition 5 (a) make use of the classification of all finite 3-homogeneous permutation groups, which itself relies on the finite simple group classification (cf. [8,19,32,36,38]). The list of groups which have to be examined is as follows:

Let  $G$  be a finite 3-homogeneous permutation group on a set  $X$  with  $|X| \geq 4$ . Then  $G$  is either of

**(A) Affine Type:**  $G$  contains a regular normal subgroup  $T$  which is elementary Abelian of order  $v = 2^d$ . If we identify  $G$  with a group of affine transformations

$$x \mapsto x^g + u$$

of  $V = V(d, 2)$ , where  $g \in G_0$  and  $u \in V$ , then particularly one of the following occurs:

- (1)  $G \cong AGL(1, 8)$ ,  $A\Gamma L(1, 8)$ , or  $A\Gamma L(1, 32)$
- (2)  $G_0 \cong SL(d, 2)$ ,  $d \geq 2$
- (3)  $G_0 \cong A_7$ ,  $v = 2^4$

or

**(B) Almost Simple Type:**  $G$  contains a simple normal subgroup  $N$ , and  $N \leq G \leq \text{Aut}(N)$ . In particular, one of the following holds, where  $N$  and  $v = |X|$  are given as follows:

- (1)  $A_v$ ,  $v \geq 5$
- (2)  $PSL(2, q)$ ,  $q > 3$ ,  $v = q + 1$
- (3)  $M_v$ ,  $v = 11, 12, 22, 23, 24$  (Mathieu groups)
- (4)  $M_{11}$ ,  $v = 12$

We note that if  $q$  is odd, then  $PSL(2, q)$  is 3-homogeneous for  $q \equiv 3 \pmod{4}$ , but not for  $q \equiv 1 \pmod{4}$ , and hence not every group  $G$  of almost simple type satisfying (2) is 3-homogeneous on  $X$ .

• If  $G \leq \text{Aut}(\mathcal{D})$  acts block-transitively on any Steiner  $t$ -design  $\mathcal{D}$  with  $t \geq 6$ , then in particular  $G$  acts point 2-transitively on  $\mathcal{D}$  by Proposition 5 (a). Applying Lemma 1 (b) yields the equation

$$b = \frac{\binom{v}{t}}{\binom{k}{t}} = \frac{v(v-1)|G_{xy}|}{|G_B|},$$

where  $x$  and  $y$  are two distinct points in  $X$  and  $B$  is a block in  $\mathcal{B}$ . Combined with the combinatorial properties in Sect. 3, this arithmetical condition yields in some of the cases under consideration immediately strong results. In other cases, particular Diophantine equations arise which have to be examined in more detail.

• As for the flag-transitive treatment (cf. [25,26]), the projective group containing  $PSL(2, q)$  – although group-theoretically well understood – requires some complicated analysis in this context. This includes a detailed consideration of the orbit-lengths from the action of subgroups of  $PSL(2, q)$  on the points of the projective line. The cases excluded from the theorem remain elusive. However, it seems to be very unlikely that admissible parameter sets of Steiner 6-designs can be found in view of the arithmetical conditions that are imposed in these cases.

## References

1. Beth, T., Jungnickel, D., Lenz, H.: Design Theory. In: Encyclopedia of Math. and Its Applications 69/78, vol. I and II. Cambridge Univ. Press, Cambridge (1999)
2. Betten, A.: Genealogy of  $t$ -designs, Australas. J. Comb. 29, 3–34 (2004)
3. Block, R.E.: Transitive groups of collineations on certain designs. Pacific J. Math. 15, 13–18 (1965)
4. Buekenhout, F.: Remarques sur l'homogénéité des espaces linéaires et des systèmes de blocs. Math. Z. 104, 144–146 (1968)
5. Buekenhout, F. (ed.): Handbook of Incidence Geometry. North-Holland, Amsterdam (1995)
6. Buekenhout, F., Delandtsheer, A., Doyen, J., Kleidman, P.B., Liebeck, M.W., Saxl, J.: Linear spaces with flag-transitive automorphism groups. Geom. Dedicata 36, 89–94 (1990)
7. Cameron, P.J.: Parallelisms of Complete Designs. London Math. Soc. Lecture Note Series, vol. 23. Cambridge Univ. Press, Cambridge (1976)
8. Cameron, P.J.: Finite permutation groups and finite simple groups. Bull. London Math. Soc. 13, 1–22 (1981)

9. Cameron, P.J.: *Permutation Groups*. Cambridge Univ. Press, Cambridge (1999)
10. Cameron, P.J., Praeger, C.E.: Block-transitive  $t$ -designs, II: large  $t$ . In: De Clerck, F., et al. (eds.) *Finite Geometry and Combinatorics (Deinze 1992)*. London Math. Soc. Lecture Note Series, vol. 191, pp. 103–119. Cambridge Univ. Press, Cambridge (1993)
11. Cameron, P.J., van Lint, J.H.: *Designs, Graphs, Codes and their Links*. Cambridge Univ. Press, Cambridge (1991)
12. Carmichael, R.D.: *Introduction to the Theory of Groups of Finite Order*, Ginn, Boston (1937); Reprint: Dover Publications, New York (1956)
13. Colbourn, C.J., Dinitz, J.H. (eds.): *Handbook of Combinatorial Designs*, 2nd edn. CRC Press, Boca Raton (2006)
14. Conway, J.H., Sloane, N.J.A.: *Sphere Packings, Lattices and Groups*, 3rd edn. Springer, Heidelberg (1998)
15. Delandtsheer, A.: Finite flag-transitive linear spaces with alternating socle. In: Betten, A., et al. (eds.) *Algebraic Combinatorics and Applications, Proc. Euroconf. (Gößweinstein 1999)*, pp. 79–88. Springer, Berlin (2001)
16. Dembowski, P.: *Finite Geometries*. Springer, Heidelberg (1968); (Reprint 1997)
17. Dixon, J.D., Mortimer, B.: *Permutation Groups*. Springer, Heidelberg (1996)
18. Fisher, R.A.: An examination of the different possible solutions of a problem in incomplete blocks. *Ann. Eugenics* 10, 52–75 (1940)
19. Gorenstein, D.: *Finite Simple Groups. An Introduction to Their Classification*. Plenum Publishing Corp., New York (1982)
20. Hall Jr., M.: *Combinatorial Theory*, 2nd edn. J. Wiley, New York (1986)
21. Higman, D.G., McLaughlin, J.E.: Geometric ABA-groups. *Illinois J. Math.* 5, 382–397 (1961)
22. Huber, M.: Classification of flag-transitive Steiner quadruple systems. *J. Combin. Theory, Series A* 94, 180–190 (2001)
23. Huber, M.: The classification of flag-transitive Steiner 3-designs. *Adv. Geom.* 5, 195–221 (2005)
24. Huber, M.: *On Highly Symmetric Combinatorial Designs*, Habilitationsschrift, Univ. Tübingen (2005), Shaker Verlag, Aachen (2006)
25. Huber, M.: The classification of flag-transitive Steiner 4-designs. *J. Algebr. Comb.* 26, 183–207 (2007)
26. Huber, M.: A census of highly symmetric combinatorial designs. *J. Algebr. Comb.* 26, 453–476 (2007)
27. Huber, M.: Coding theory and algebraic combinatorics. In: Woungang, I., et al. (eds.) *Selected Topics in Information and Coding Theory*, 33 p. World Scientific, Singapore (to appear)
28. Huber, M.: *Flag-transitive Steiner Designs*, Birkhäuser, Basel, Berlin, Boston (to appear)
29. Huffman, W.C., Pless, V. (eds.): *Handbook of Coding Theory*, vol. I and II. North-Holland, Amsterdam (1998)
30. Huffman, W.C., Pless, V.: *Fundamentals of Error-Correcting Codes*. Cambridge Univ. Press, Cambridge (2003)
31. Hughes, D.R., Piper, F.C.: *Design Theory*. Cambridge Univ. Press, Cambridge (1985)
32. Kantor, W.M.:  $k$ -homogeneous groups. *Math. Z.* 124, 261–265 (1972)
33. Kantor, W.M.: 2-transitive and flag-transitive designs. In: Jungnickel, D., et al. (eds.) *Coding Theory, Design Theory, Group Theory, Proc. Marshall Hall Conf. (Burlington, VT, 1990)*, pp. 13–30. J. Wiley, New York (1993)

34. Kaski, P., Östergård, P.R.J.: *Classification Algorithms for Codes and Designs*. Springer, Heidelberg (2006)
35. Kleidman, P.B.: The finite flag-transitive linear spaces with an exceptional automorphism group. In: Kramer, E.S., Magliveras, S.S. (eds.) *Finite Geometries and Combinatorial Designs* (Lincoln, NE, 1987), vol. 111, pp. 117–136, *Contemp. Math.* Amer. Math. Soc., Providence, RI (1990)
36. Liebeck, M.W.: The affine permutation groups of rank three. *Proc. London Math. Soc.* 54(3), 477–516 (1987)
37. Liebeck, M.W.: The classification of finite linear spaces with flag-transitive automorphism groups of affine type. *J. Combin. Theory, Series A* 84, 196–235 (1998)
38. Livingstone, D., Wagner, A.: Transitivity of finite permutation groups on unordered sets. *Math. Z.* 90, 393–403 (1965)
39. Lüneburg, H.: Fahnenhomogene Quadrupelsysteme. *Math. Z.* 89, 82–90 (1965)
40. Pei, D.: *Authentication Codes and Combinatorial Designs*. CRC Press, Boca Raton (2006)
41. Ray-Chaudhuri, D.K., Wilson, R.M.: On  $t$ -designs. *Osaka J. Math.* 12, 737–744 (1975)
42. Rota, G.-C.: On the foundations of combinatorial theory I: theory of Möbius functions. *Z. Wahrscheinlichkeitsrechnung u. verw. Geb.* 2, 340–368 (1964)
43. Saxl, J.: On finite linear spaces with almost simple flag-transitive automorphism groups. *J. Combin. Theory, Series A* 100, 322–348 (2002)
44. Stinson, D.R.: *Combinatorial Designs: Constructions and Analysis*. Springer, Heidelberg (2004)
45. Stinson, D.R.: *Cryptography*, 3rd edn. CRC Press, Boca Raton (2005)
46. Teirlinck, L.: Non-trivial  $t$ -designs without repeated blocks exist for all  $t$ . *Discrete Math.* 65, 301–311 (1987)
47. Tits, J.: Sur les systèmes de Steiner associés aux trois “grands” groupes de Mathieu. *Rendic. Math.* 23, 166–184 (1964)
48. Tonchev, V.D.: *Combinatorial Configurations: Designs, Codes, Graphs*. Longman, Harlow (1988)
49. Tuan, N.D.: Simple non-trivial designs with an arbitrary automorphism group. *J. Combin. Theory, Series A* 100, 403–408 (2002)
50. Wielandt, H.: *Finite Permutation Groups*. Academic Press, London (1964)

# New Spatial Configurations

Harald Gropp

Mühlingstr.19, D-69121 Heidelberg, Germany  
d12@ix.urz.uni-heidelberg.de

**Abstract.** This second paper on constructions of spatial configurations follows the author's paper of 1994 [2]. For the first time again new spatial configurations are constructed, in particular a configuration  $(33_8)_2$ , the smallest known configuration  $(v_8)_2$ , and several configurations  $(v_9)_2$ , in particular for  $v = 40$  and for  $v \geq 43$ .

## 1 Introduction and Notation

The most important definitions and notations are repeated here from [2]. For further details the reader is referred to this paper and to [1] where the connection to possible applications like radio astronomy and engineering sciences is given.

**Definition 1.1.** A  $\lambda$ -configuration  $(v_r, b_k)_\lambda$  is a finite incidence structure consisting of a set of points and a set of subsets ( called lines ) of this set such that

1. there are  $v$  points and  $b$  lines,
2. there are  $k$  points on each line and  $r$  lines through each point,
3. two different points are connected by at most  $\lambda$  lines and two different lines intersect each other in at most  $\lambda$  points.

For  $\lambda = 1$  we obtain 1-configurations or just configurations.

For  $\lambda = 2$  we obtain 2-configurations or spatial configurations.

Analogously to the case of 1-configurations the following multigraph can be defined.

**Definition 1.2.** The configuration multigraph of a 2-configuration  $(v_r, b_k)_2$  has as vertex set the set of  $v$  points. There is a double edge between two vertices iff the two points are not on a common line of the configuration. There is a simple edge iff the two points are on a unique common line. There is no edge iff the two points are on exactly two common lines.

### Remark 1.3

1. Since the dual structure of a 2-configuration is also a 2-configuration it can be assumed that  $b \geq v$ .
2. Necessary conditions for the existence of a 2-configuration  $(v_r, b_k)_2$  are  $vr = bk$  and  $v \geq 1 + r(k - 1)/2$ .
3. The configuration multigraph is regular of degree  $d$ .  $d = 2(v - 1) - r(k - 1)$  is called the deficiency of the 2-configuration.

In this paper symmetric 2-configurations ( i.e.  $v = b$  and hence  $r = k$  ) will be discussed and denoted by  $(v_k)_2$  instead of  $(v_k, v_k)_2$ .

Concerning the relation of spatial configurations to usual configurations the reader is referred to an author's handbook article on configurations [3].

## 2 New Constructions

The solutions are given as difference triangles where the first row is the base row and the  $i^{th}$  row contains all the sums of  $i$  consecutive numbers of the base row. No number must occur more than twice and  $v/2$  must occur at most once (all numbers are considered mod  $(v)$ ). A base block of the 2-configuration is obtained by taking 0 and the first entry of each row. All blocks are then constructed by developing this block mod  $(v)$ , i.e. adding 1 to each element (the second block), adding 2 (the third block), ... and finally adding  $v - 1$  (the last block).

### 2.1 $k \leq 7$

All existence problems for  $k \leq 6$  were solved in [2].

For  $k = 7$  all configurations  $(v_7)_2$  with  $v \geq 24$  are constructed in [2]. The nonexistence of a configuration  $(22_7)_2$  is a consequence of the theorem of Bruck-Ryser-Chowla. Such a configuration would be a biplane  $(22,7,2)$  (see any book on design theory). It is mentioned in a recent paper by Kaski and Östergård [4] that there is no configuration  $(23_7)_2$ .

### 2.2 $k = 8$

For  $k = 8$  the following difference triangle, already obtained in [2], implies the existence of 2-configurations  $(v_8)_2$  for all  $v \geq 34$ .

$$\begin{array}{cccccc}
 5 & 3 & 3 & 1 & 5 & 2 & 2 \\
 & 8 & 6 & 4 & 6 & 7 & 4 \\
 & & 11 & 7 & 9 & 8 & 9 \\
 & & & 12 & 12 & 11 & 10 \\
 & & & & 17 & 14 & 13 \\
 & & & & & 19 & 16 \\
 & & & & & & 21
 \end{array}$$

The following newly constructed triangle yields a 2-configuration  $(33_8)_2$ , up to now the smallest 2-configuration with  $k = 8$ .

$$\begin{array}{cccccc}
 2 & 6 & 4 & 3 & 1 & 1 & 5 \\
 & 8 & 10 & 7 & 4 & 2 & 6 \\
 & & 12 & 13 & 8 & 5 & 7 \\
 & & & 15 & 14 & 9 & 10 \\
 & & & & 16 & 15 & 14 \\
 & & & & & 17 & 20 \\
 & & & & & & 22
 \end{array}$$

**Theorem 2.1** *There is a configuration  $(v_8)_2$  for all  $v \geq 33$ . For  $30 \leq v \leq 32$  the existence is in doubt. There is no configuration  $(29_8)_2$ .*

**2.3  $k = 9$**

2	1	4	6	3	1	5	2
3	5	10	9	4	6	7	
7	11	13	10	9	8		
13	14	14	15	11			
16	15	19	17				
17	20	21					
22	22						
24							

The above triangle yields the existence of a configuration  $(40_9)_2$ .

3	1	4	2	5	1	10	3
4	5	6	7	6	11	13	
8	7	11	8	16	14		
10	12	12	18	19			
15	13	22	21				
16	23	25					
26	26						
29							

This triangle implies the existence of a configuration  $(43_9)_2$ .

3	2	1	9	1	6	2	5
5	3	10	10	7	8	7	
6	12	11	16	9	13		
15	13	17	18	14			
16	19	19	23				
22	21	24					
24	26						
29							

This triangle yields the existence of a configuration  $(44_9)_2$ .

1	5	2	2	8	1	3	3
6	7	4	10	9	4	6	
8	9	12	11	12	7		
10	17	13	14	15			
18	18	16	17				
19	21	19					
22	24						
25							

This triangle yields the existence of a configuration  $(v_9)_2$  for all  $v \geq 45$ .

**Theorem 2.2** *There is a configuration  $(v_9)_2$  for  $v = 37, 38, 40$  and all  $v \geq 43$ . For  $v = 39, 41, 42$  the existence is in doubt.*

**Remark 2.3** *The existence of a configuration  $(37_9)_2$  ( a biplane) and a configuration  $(38_9)_2$  (a semiplane) was discussed in [2].*

### 3 The Existence Table of 2-Configurations

In this last section the results are summarized and exhibited in an updated table.

An entry in plain type means that the corresponding configuration exists.

**Bold configurations** do not exist.

A blank space shows that the existence problem is open.

Deficiency	0	2	4	6	8	10	12	14
$k$								
3	4 <sub>3</sub>	5 <sub>3</sub>	6 <sub>3</sub>	7 <sub>3</sub>	8 <sub>3</sub>	9 <sub>3</sub>	10 <sub>3</sub>	11 <sub>3</sub>
4	7 <sub>4</sub>	8 <sub>4</sub>	9 <sub>4</sub>	10 <sub>4</sub>	11 <sub>4</sub>	12 <sub>4</sub>	13 <sub>4</sub>	14 <sub>4</sub>
5	11 <sub>5</sub>	12 <sub>5</sub>	13 <sub>5</sub>	14 <sub>5</sub>	15 <sub>5</sub>	16 <sub>5</sub>	17 <sub>5</sub>	18 <sub>5</sub>
6	16 <sub>6</sub>	17 <sub>6</sub>	18 <sub>6</sub>	19 <sub>6</sub>	20 <sub>6</sub>	21 <sub>6</sub>	22 <sub>6</sub>	23 <sub>6</sub>
7	<b>22<sub>7</sub></b>	<b>23<sub>7</sub></b>	24 <sub>7</sub>	25 <sub>7</sub>	26 <sub>7</sub>	27 <sub>7</sub>	28 <sub>7</sub>	29 <sub>7</sub>
8	<b>29<sub>8</sub></b>				33 <sub>8</sub>	34 <sub>8</sub>	35 <sub>8</sub>	36 <sub>8</sub>
9	37 <sub>9</sub>	38 <sub>9</sub>	40 <sub>9</sub>				43 <sub>9</sub>	44 <sub>9</sub>

### References

1. Gropp, H.: On Golomb birulers and their applications. *Mathematica Slovaca* 42, 517–529 (1992)
2. Gropp, H.: On symmetric spatial configurations. *Discrete Math.* 125, 201–209 (1994)
3. Gropp, H.: Configurations. In: Colbourn, C.J., Dinitz, J.H. (eds.) *Handbook of Combinatorial Designs*, 2nd edn., Boca Raton, pp. 352–354 (2007)
4. Kaski, P., Östergård, P.: There exists no symmetric configuration with 33 points and line size 6. *Australasian Journal of Combinatorics* 38, 273–277 (2007)

# Construction of Large Constant Dimension Codes with a Prescribed Minimum Distance

Axel Kohnert and Sascha Kurz

Department of Mathematics  
University of Bayreuth  
95440 Bayreuth  
Germany  
axel.kohnert@uni-bayreuth.de,  
sascha.kurz@uni-bayreuth.de

**Abstract.** In this paper we construct constant dimension codes with prescribed minimum distance. There is an increased interest in subspace codes in general since a paper [13] by Kötter and Kschischang where they gave an application in network coding. There is also a connection to the theory of designs over finite fields. We will modify a method of Braun, Kerber and Laue [7] which they used for the construction of designs over finite fields to construct constant dimension codes. Using this approach we found many new constant dimension codes with a larger number of codewords than previously known codes. We finally give a table of the best constant dimension codes we found.

**Keywords:** network coding,  $q$ -analogue of Steiner systems, subspace codes.

## 1 Introduction

### 1.1 Subspace Codes

In [13] R. Kötter and F. R. Kschischang developed the theory of subspace codes for applications in network coding. We will recapitulate their definitions in a slightly different manner. We denote by  $L(GF(q)^v)$  the lattice of all subspaces of the space of dimension  $v$  over the finite field with  $q$  elements together with the partial order is given by inclusion. A *subspace code*  $C$  is a subset of  $L(GF(q)^v)$ . If all the subspaces in  $C$  are of the same dimension then  $C$  is a *constant dimension code*.

The subspace distance between two spaces  $V$  and  $W$  in  $L(GF(q)^v)$  is defined as

$$d_S(V, W) := \dim(V + W) - \dim(V \cap W)$$

which is equal to

$$\dim(V) + \dim(W) - 2 \dim(V \cap W).$$

This defines a metric on  $L(GF(q)^v)$ . The minimum (subspace) distance of a subspace code  $C$  is defined as

$$D_S(C) := \min\{d_S(V, W) : V, W \in C \text{ and } V \neq W\}.$$

We define now the optimal (subspace) code problem:

(P1) For a given lattice  $L(GF(q)^v)$  (based on inclusion) fix a minimum (subspace) distance  $d$  and find the maximal number  $m$  of subspaces  $V_1, \dots, V_m$  in  $L(GF(q)^v)$  such that the corresponding subspace code  $C = \{V_1, \dots, V_m\}$  has at least minimum distance  $d$ .

The following point of view is useful for the study of subspace codes: We first define the *Hamming graph* with parameters  $v$  and  $q$  by taking as vertex-set the words of length  $v$  over the alphabet  $GF(q)$  and connecting two vertices  $u, w$  by an edge if the minimum distance between  $u$  and  $w$  is equal to one. One of the classical problems in coding theory can then be stated as follows:

(P2) Given the Hamming graph of all words of length  $v$  and a minimum distance  $d$  find a maximal number  $m$  of words such that the pairwise minimum distance is at least  $d$ .

If we substitute the Hamming graph by the *Hasse diagram* of  $L(GF(q)^v)$  (vertices are the subspaces of  $GF(q)^v$  and two subspaces are connected by an edge if they are direct neighbors in the partial order arising from inclusion) the problem (P2) becomes problem (P1). Both problems are special cases of a packing problem in a graph. If we start with problem (P2) and use the 'field with one element' we get problem (P1). Because of this property we say (P2) is the  $q$ -analogue of (P1). This connection is well known (e.g. [1,17]) and will be useful in the following. Since the publication of the paper by Kötter and Kschischang the constant dimension codes as the  $q$ -analogue of the constant weight codes were studied in a series of papers [10,12,23].

## 1.2 $q$ -Analogues of Designs

A  $t - (v, k, \lambda)$  design is a set  $C$  of  $k$ -element subsets (called blocks) of the set  $\{1, \dots, v\}$  such that each  $t$ -element subset of  $\{1, \dots, v\}$  appears in exactly  $\lambda$  blocks. The special case of  $\lambda = 1$  is called a Steiner system.

The same construction which was used to connect problem (P1) to (P2) in the subsection above can be used to define the  $q$ -analogue of a  $t$ -design. A  $t - (v, k, \lambda)$  design over the finite field  $GF(q)$  is a multiset  $C$  of  $k$ -dimensional subspaces (called  $q$ -blocks) of the  $v$ -dimensional vector space  $GF(q)^v$  such that each  $t$ -dimensional subspace of  $GF(q)^v$  is a subspace of exactly  $\lambda$   $q$ -blocks.

The connection with the constant dimension codes is given by the following observation in the case of a  $q$ -analogue of a Steiner system: Given a  $q$ -analogue of a  $t - (v, k, 1)$  design  $C$  we get a constant dimension code of minimum distance  $2(k - t + 1)$ . As each  $t$ -dimensional space is contained in exactly one  $k$ -dimensional subspace the intersection between two spaces from  $C$  is at most  $(t - 1)$ -dimensional. Therefore the minimum distance of  $C$  is at least  $2(k - t + 1)$ . On the other hand given any  $(t - 1)$ -dimensional subspace  $V$  we can find two  $t$ -dimensional spaces  $U, W$  with intersection  $V$  and then two unique  $q$ -blocks containing  $U$  and  $W$ . The minimum distance between these  $q$ -blocks is  $2(k - t + 1)$ .

$q$ -analogues of designs were introduced by Thomas in 1987 [19]. Later they were studied in a paper by Braun et al. [7] where the authors constructed the first non-trivial  $q$ -analogue of a 3-design. We will use the methods described in their paper to construct constant dimension codes.

In later papers by Thomas [20] and Etzion and Schwartz [17] it was shown that there are severe restrictions on the possible existence of  $q$ -analogues of Steiner systems. We will search for a collection of subspaces satisfying only the conditions given by (P1) and not for the stronger condition satisfied by a  $q$ -analogue of a Steiner system. But in general the methods described in this paper can also be used for the search for Steiner systems.

## 2 Construction of Constant Dimension Codes

In this section we describe how to construct a constant dimension code  $C$  using a system of Diophantine linear equations and inequalities. Due to the definition of the subspace distance for all  $V, W \in C$  we have  $d_S(V, W) = 2k - 2\dim(V \cap W)$  where  $k$  is the dimension of the code. Thus the minimum subspace distance has to be an even number less or equal to  $2k$ . To construct a constant dimension code of dimension  $k$  and minimum subspace distance  $2d$  we have to find  $n$  subspaces  $\{V_1, \dots, V_n\}$  of dimension  $k$  such that there is no subspace of dimension  $k - d + 1$  contained in two of the selected  $k$ -spaces. We define  $M$  as the incidence matrix of the incidence system between the  $(k - d + 1)$ -spaces (labeling the rows of  $M$ ) and the  $k$ -spaces (labeling the columns):

$$M_{W,V} := \begin{cases} 1 & \text{if } V \text{ contains } W, \\ 0 & \text{otherwise.} \end{cases}$$

Using  $M$  we get the description of a constant dimension code as the solution of a Diophantine system. We denote by  $s$  the number of columns of  $M$ .

### Theorem 1

*There is a constant dimension code with  $m$  codewords and minimum distance at least  $2d$  if and only if there is a  $(0/1)$ -solution  $x = (x_1, \dots, x_s)^T$  of the following system of one equation and a set of inequalities:*

$$\sum_{i=1}^s x_i = m \tag{1}$$

$$Mx \leq \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}. \tag{2}$$

This set of inequalities has to be read as follows: A solution  $x$  has the property that the product of  $x$  with a single row of  $M$  is 0 or 1. Otherwise if the inner product of  $x$  with the row labeled by  $W$  is larger than one, then the subspace  $W$  is contained in more than one subspaces  $V$ . To get the constant dimension code corresponding to a solution we have to use the  $(0/1)$ -vector  $x$  as the characteristic vector of a subset of the set

of all  $k$ -dimensional subspaces of  $GF(q)^v$ . Theorem 1 is a generalization of the Diophantine system describing the search for a  $q$ -analogue of a Steiner system which was given in [7].

**Corollary 1.** [7]

*There is a  $q$ -analogue of a  $(k - d + 1) - (v, k, 1)$  design with  $b$  blocks if and only if there is a  $(0/1)$ -solution  $x = (x_1, \dots, x_s)^T$  of the following system of Diophantine linear equations:*

$$\sum_{i=1}^s x_i = b \tag{3}$$

$$Mx = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}. \tag{4}$$

The size of these problems is given by the number of subspaces in  $GF(q)^v$ . In general this number is growing too fast. The number of  $k$ -dimensional subspaces of  $GF(q)^v$  is given by the  $q$ -binomial coefficients:

$$\begin{bmatrix} v \\ k \end{bmatrix}_q := \prod_{j=1..k} \frac{(1 - q^{v+1-j})}{(1 - q^j)}.$$

Already in the smallest case of a 2-analogue of the Fano plane ( $v = 7, k = 3, d = 2$ ) the matrix  $M$  has 11811 columns and 2667 rows.

### 3 Constant Dimension Codes with Prescribed Automorphisms

To handle also larger cases we apply the following method. We no longer look for an arbitrary constant dimension code. We are now only interested in a set of spaces which have a prescribed group of automorphisms. An automorphism  $\varphi$  of set  $C = \{V_1, \dots, V_m\}$  is an element from  $GL(v, GF(q))$  such that  $C = \{\varphi(V_1), \dots, \varphi(V_m)\}$ . We denote by  $G$  the group of automorphisms of  $C$ , which is a subgroup of  $GL(v, GF(q))$ .

The main advantage of prescribing automorphisms is that the size of the system of equations is much smaller. The number of variables will be the number of orbits of  $G$  on the  $k$ -spaces. The number of equations or inequalities will be the number of orbits on the  $(k - d + 1)$ -spaces. The construction process will then have two steps:

- In a first step the solution of a construction problem is described as a solution of a Diophantine system of linear equations.
- In a second step the size of the linear system is reduced by prescribing automorphisms.

This construction method is a general approach that works for many discrete structures as designs [3,14],  $q$ -analogs of designs [6,7], arcs in projective geometries [8], linear codes [2,4,5,15] or quantum codes [21].

The general method is as follows: The matrix  $M$  is reduced by adding up columns (labeled by the  $k$ -spaces) corresponding to the orbits of  $G$ . Now because of the relation

$$W \text{ subspace of } V \iff \varphi(W) \text{ subspace of } \varphi(V) \quad (5)$$

for any  $k$ -space  $V$  and  $(k-d)$ -space  $W$  and any automorphism  $\varphi \in G$  the rows corresponding to lines in an orbit under  $G$  are equal. Therefore the redundant rows are removed from the system of equations and we get a smaller matrix denoted by  $M^G$ . The number of rows of  $M^G$  is then the number of orbits of  $G$  on the  $(k-d+1)$ -spaces. The number of columns of  $M^G$  is the number of orbits of  $G$  on the  $k$ -spaces. We denote by  $\omega_1, \dots$  the orbits on the  $k$ -spaces and by  $\Omega_1, \dots$  the orbits on the  $(k-d+1)$ -spaces. For an entry of  $M^G$  we have:

$$M_{\Omega_i, \omega_j}^G = |\{V \in \omega_j : W \text{ is a subspace of } V\}|$$

where  $W$  is a representative of the orbit  $\Omega_i$  of  $(k-d+1)$ -spaces. Because of property (5) the matrix  $M$  is well-defined as the definition of  $M_{\Omega_i, \omega_j}^G$  is independent of the representative  $W$ . Now we can restate the above theorem in a version with the condensed matrix  $M^G$ :

## Theorem 2

Let  $G$  be a subgroup of  $GL(v, GF(q))$ . There is a constant dimension code of length  $m$  and minimum distance at least  $2d$  whose group of automorphisms contains  $G$  as a subgroup if, and only if, there is a  $(0/1)$ -solution  $x = (x_1, \dots)^T$  of the following system of one equation and a set of inequalities:

$$\sum_i |\omega_i| x_i = m \quad (6)$$

$$M^G x \leq \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}. \quad (7)$$

There is one further reduction possible. We are looking for a  $(0/1)$ -solution where each inner product of a row of  $M^G$  and the vector  $x$  is less or equal to 1. We can remove columns of  $M^G$  with entries greater than 1. This gives a further reduction of the size of  $M^G$ . After this last removal of columns we again check on equal rows and also on rows containing only entries equal to zero. We remove these all zero rows and all but one copy of the equal rows.

In order to locate large constant dimension codes with given parameters  $q$ ,  $k$  and  $2d$  we try to find feasible solutions  $x = (x_1, \dots)^T$  of the system of equations of Theorem 3 for a suitable chosen group  $G$  and a suitable chosen length  $m$ . Here we remark that we have the freedom to change equation (6) of Theorem 2 into

$$\sum_i |\omega_i| x_i \geq m.$$

For this final step we use some software. Currently we use a variant of an LLL based solver written by Alfred Wassermann [22] or a program by Johannes Zwanzger [24]

which uses some heuristics especially developed for applications in coding theory. The advantage of the LLL based solver is that we definitely know whether there exist feasible solutions or not whenever the program runs long enough to terminate. Unfortunately for the examples of Section 5 this never happens so that practically we could only use this solver as a heuristic to quickly find feasible solutions.

If we change equation (6) into a target function

$$f(x) = f(x_1, \dots) = \sum_i |\omega_i| x_i$$

we obtain a formulation as a binary linear optimization problem. In this case we can apply the commercial ILOG CPLEX 11.1.0 software for integer linear programs. The big advantage of this approach is that at every time of the solution process we have lower bounds, corresponding to a feasible solution with the largest  $f(x)$ -value found so far, and upper bounds on  $f(x)$ .

We can even reformulate our optimization problem in the language of graph theory. Here we consider the variable indices  $i$  as vertices of a graph  $\mathcal{G}$  each having weight  $|\omega_i|$ . The edges of  $\mathcal{G}$  are implicitly given by inequality (7). Therefore let us denote the  $i$ th row of  $M^G$  by  $M_{i,\cdot}^G$ . Now the inequality  $M_{i,\cdot}^G \leq 1$  translates into the condition that the set

$$\mathcal{C}_i := \{j : M_{i,j}^G = 1\}$$

is an independence set in  $\mathcal{G}$ . To construct the graph  $\mathcal{G}$  we start with a complete graph and for each row  $M_{i,\cdot}^G$  we delete all edges between vertices in  $\mathcal{C}_i$ . Now an optimal solution of the binary linear program corresponds to a maximum weight clique in  $\mathcal{G}$ . Again there exist heuristics and exact algorithms to determine maximum weight cliques in graphs. An available software package for this purpose is e.g. CLIQUER [16].

This approach allows to use clique bounds from algebraic graph theory to obtain upper bounds on the target function  $f(x)$ . In the case where we are able to locate large independent sets in  $\mathcal{G}$  which are not subsets of the  $\mathcal{C}_i$  we can use them to add further inequalities to (7). If those independent sets are large enough and not too many then a solver for integer linear programs highly benefits from the corresponding additional inequalities.

For theoretical upper bounds and practical reasons how to quickly or exhaustively locate solutions of our system of Theorem 2 it is very useful to have different formulations of our problem to be able to apply different solvers.

### 3.1 Example

We start with the space  $GF(2)^7$ . We now describe the construction of a subspace code with 304 codewords and constant dimension equal to 3. This code will have minimum subspace distance 4. The matrix  $M$  is the incidence matrix between the 3-dimensional subspaces of  $GF(2)^7$  and the 2-dimensional subspaces. Without further reductions this matrix has  $\begin{bmatrix} 7 \\ 3 \end{bmatrix}_2 = 11811$  columns and  $\begin{bmatrix} 7 \\ 2 \end{bmatrix}_2 = 2667$  rows. We prescribe now a group  $G$  of automorphisms generated by a single element:



the Singer subgroup  $G$ . Adding one to each number results in the complete orbit by performing it  $l$  times.

*Example 1.*  $q = 2, v = 5, k = 2$

A two-dimensional binary subspace contains three one-dimensional subspaces. We get a two-dimensional space by taking the two one-dimensional spaces labeled  $\{0, 1\}$  and the third one given by the linear combination of these two will have a certain number, in this example  $\{14\}$ . Therefore we have a two dimensional space described by the three numbers  $\{0, 1, 14\}$ . To get the complete orbit under the Singer subgroup we simply have to increase the numbers by one for each multiplication by the generator  $g$  of the Singer subgroup. The orbit length of the Singer subgroup is 31 and the orbit is built by the 31 sets:  $\{0, 1, 14\}, \{1, 2, 15\}, \dots, \{16, 17, 30\}, \{0, 17, 18\}, \dots, \{12, 29, 30\}, \{0, 13, 30\}$ .

To describe the different orbits of the Singer subgroup we build the following set of pairwise distances:

Let  $s$  be the number of one-dimensional subspaces in  $k$ -space. Let  $\{v_1, \dots, v_s\} \subset \{0, 1, \dots, l\}$  be the set of  $s$  numbers describing a fixed  $k$ -space  $U$ . Denote by  $d_{\{i,j\}}$  the distance between the two numbers  $v_i$  and  $v_j$  modulo the length of the Singer cycle.  $d_{\{i,j\}}$  is a number between 1 and  $l/2$ . We define the multiset  $D_U := \{d_{\{i,j\}} : 1 \leq i < j \leq s\}$ . We call  $D_U$  the distance distribution of the subspace  $U$ . All the spaces in an orbit of a Singer subgroup have the same distance distribution and on the other hand different orbits have different distance distribution. We therefore also say that  $D_U$  is the distance distribution of the orbit.

We use these distance distribution to label the different orbits of the Singer subgroup of the  $k$ -spaces. The first observation is:

**Lemma 1.** *A Singer orbit as a subspace code*

*An orbit  $C = \{V_0, \dots, V_l\}$  of a Singer subgroup on the  $k$ -subspaces of  $GF(q)^v$  is a subspace code of minimum distance  $2(k - 1)$  if and only if the distance distribution of the orbit has no repeated numbers.*

*Proof.* We have to show that the intersection of any pair of spaces in  $C$  has at most one one-dimensional space in common. Having no repeated entry in the distance distribution means that a pair of numbers (i.e. pair of one-dimensional subspaces) in a  $q$ -block  $b$  of  $C$  can not be built again by shifting the numbers in  $b$  using the operation of the Singer subgroup on  $b$ .

The same is true if we want to construct a subspace code by combining several orbits of the Singer subgroup. We have to check that the intersection between two spaces is at most one-dimensional. For this we define the matrix  $S$ , whose columns are labeled by the orbits  $\Omega_j$  of the Singer subgroup on the  $k$ -dimensional subspaces of  $GF(q)^v$  and the rows are labeled by the possible numbers  $i \in \{0, \dots, l/2\}$  in the distance distribution of the  $k$ -spaces. Denoting by  $D_{\Omega_j}$  the distance distribution of the  $j$ -th orbit, we define an entry of the matrix  $S$  by

$$S_{i,\Omega_j} := \begin{cases} 1 & \text{if } i \in D_{\Omega_j} \\ 0 & \text{otherwise.} \end{cases}$$

Using this matrix  $S$  we have the following characterization of constant dimension codes with prescribed automorphisms:

### Theorem 3

There is a constant dimension code  $C$  with  $n \cdot (l+1)$  codewords and minimum distance at least  $2(k-1)$  whose group of automorphisms contains the Singer subgroup as a subgroup if and only if there is a  $(0/1)$ -solution  $x = (x_1, \dots)^T$  of the following system of one equation and a set of inequalities:

$$\sum_i x_i = n \quad (8)$$

$$Sx \leq \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}. \quad (9)$$

This is the final system of one Diophantine linear equation together with  $l/2 + 1$  inequalities which we successfully solved in several cases.

## 5 Results

As mentioned in the introduction there is an increased interest on constant dimension codes with a large number of codewords for a given minimum subspace distance. There are (very) recent ArXiv-preprints [10,11,18] giving some constructions for those codes.

Here we restrict ourselves on the binary field  $q = 2$  and dimension  $k = 3$  and minimum subspace distance  $d_S = 4$ .

Using the approach described in Section 4 it was possible to construct constant dimension codes using the Singer cycle with the following parameters. We denote by  $n$  the number of orbits used to build a solution, by  $d_S$  we denote the minimum space distance of the corresponding constant dimension code:

$v$	$k$	$n =$ number of used orbits	total number of orbits	number of codewords	best known	$d_S = 2d$
6	3	1	19	$1 \cdot 63 = 63$	71[18]	4
7	3	2	93	$2 \cdot 127 = 254$	294	4
8	3	5	381	$5 \cdot 255 = 1275^*$	1164[18]	4
9	3	11	1542	$11 \cdot 511 = 5621^*$	4657[18]	4
10	3	21	6205	$21 \cdot 1023 = 21483^*$	18631[18]	4
11	3	39	24893	$39 \cdot 2047 = 79833^*$	74531[18]	4
12	3	77	99718	$77 \cdot 4095 = 315315^*$	298139[18]	4
13	3	141	399165	$141 \cdot 8191 = 1154931$	1192587[18]	4
14	3	255	1597245	$255 \cdot 16383 = 4177665$	4770411[18]	4

In [11] the authors defined the number  $A_q(v, d_S, k)$  as the maximal number of codewords in a constant dimension code of minimum distance  $d_S$ . They derived lower and upper bounds. We have implemented the construction method described in [18] to obtain the resulting code sizes which give the lower bounds for  $A_q(v, d_S, k)$  for  $v \geq 9$ .

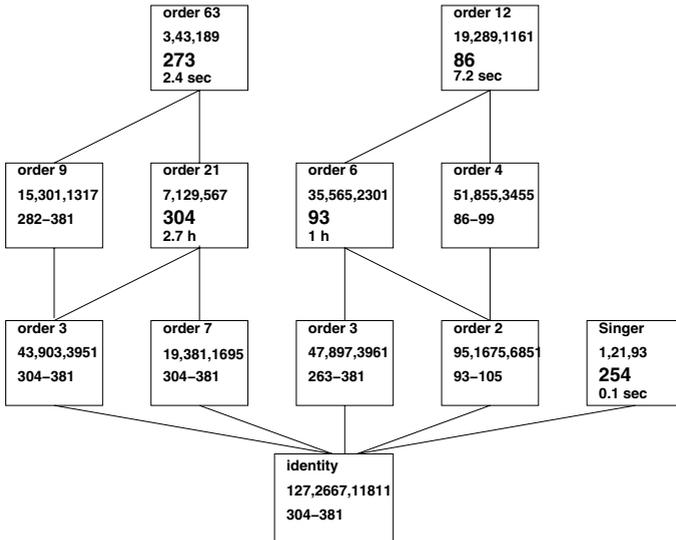
In the above table we marked codes which improved the lower bounds on  $A_q(v, d_S, k)$  with an \*. We would like to remark that for  $6 \leq v \leq 8$  our results are optimal for the Singer cycle as a subgroup of the group of automorphisms (using the formulation as a binary linear program). So far, for  $v = 9$  a code size of  $n = 12$  is theoretically possible. (In this case the corresponding binary linear program was not solved to optimality.)

Since for  $v = 6, 7$  the method using the Singer cycle was not capable of beating the best known constant dimension code we tried the more general approach described in Section 3. In both cases we improved the cardinality of the best known constant dimension codes as shown in this small table:

$v$	$k$	number of codewords	best known	$d_S = 2d$
6	3	77	71[18]	4
7	3	304	294	4

For  $v = 6$  even the original incidence matrix  $M$  or  $M^G$  where  $G$  is the identity group results in only 1395 binary variables and 651 constraints having 9765 nonzero entries. Using the ILOG CPLEX 11.1.0 solver directly on this problem yields a constant dimension code of cardinality  $n = 77$  which beats the example of [10, 18] by 6. The best known upper bound in this case is given by 81, where as the upper bound given by the linear relaxation is give by 93. Marcus Grassl (private communication) also found codes of cardinality  $n = 77$  using some heuristics together with the CLIQUER software [16].

As mentioned in Example 3.1 the original incidence matrix  $M$  is quite large. Here the direct approach has not led to any improvements. Although in general it is difficult to construct the condensed matrix  $M^G$  for an arbitrary group and larger parameters we were able to conquer the difficulties for  $v = 7, k = 3, d_S = 4$  and some groups. The group resulting in the code having 304 three-dimensional subspaces of  $GF(2)^7$  such that the intersection of two codewords has dimension at most one was already given in Example 3.1. We have tried several groups before ending up with this specific group. More details can be shown using the following diagram:



This picture shows part of the subgroup lattice of the automorphism group  $PGL(7, 2)$  of the  $L(GF(2)^7)$ . It only shows cyclic groups and in the top row we give the order of the group. In the second row we give the number of orbits on the points, lines and planes. In the third row of each entry we give the size  $lb$  of a constant dimension 6 code and the best found upper bound  $ub$  in the format  $lb - ub$ . As described in Section 3 for a given group our problem corresponds to several versions of feasibility or optimization problems. To obtain the lower bounds we have used the LLL based algorithm, the coding theoretic motivated heuristic and the ILOG CPLEX solver for integer linear programs. The upper bounds were obtained by the CPLEX solver stopping the solution process after a reasonable time. Whenever the lower and the upper bound meet we have written only one number in bold face. In each of these cases we give the necessary computation time to prove optimality in the fourth row.

As we can split orbits if we move to a subgroup we can translate a solution found for a group  $G$  into a solution for a subgroup of  $G$ . E.g. for the groups of order smaller than 21 we did not find codes of size 304 directly. This fact enables us to perform a restricted search in systems corresponding to subgroups by only considering solutions which are in some sense *near* to such a translated solution. We have tried this for the subgroups of the group of order 21 - unfortunately without success.

We would like to remark that solving the linear relaxation can prevent other heuristics from searching for good solutions where no good solutions can exist. E.g. we can calculate in a second that every code in the case of the third group in the third row can contain at most 105 codewords. Since we know better examples we can skip calculations in this group and all groups which do contain this group as a subgroup.

Finally we draw the conclusion that following the approach described in Section 3 it is indeed possible to construct good constant dimension codes for given minimum subspace distance. Prescribing the Singer cycle as a subgroup of the automorphism group has some computational advantages. The resulting codes are quite competitive for  $v \geq 8$ . The discovered constant dimension codes for  $v = 6, 7$  show that it pays off to put some effort in the calculation of the condensed matrix  $M^G$  for other groups.

## References

1. Ahlswede, R., Aydinian, H.K., Khachatrian, L.H.: On perfect codes and related concepts. *Des. Codes Cryptography* 22(3), 221–237 (2001)
2. Betten, A., Braun, M., Fripertinger, H., Kerber, A., Kohnert, A., Wassermann, A.: Error-correcting linear codes. Classification by isometry and applications. With CD-ROM. In: *Algorithms and Computation in Mathematics* 18, p. xxix, 798. Springer, Berlin (2006)
3. Betten, A., Kerber, A., Kohnert, A., Laue, R., Wassermann, A.: The discovery of simple 7-designs with automorphism group  $PT L(2,32)$ . In: Giusti, M., Cohen, G., Mora, T. (eds.) *AAECC 1995*. LNCS, vol. 948, pp. 131–145. Springer, Heidelberg (1995)
4. Braun, M.: Construction of linear codes with large minimum distance. *IEEE Transactions on Information Theory* 50(8), 1687–1691 (2004)
5. Braun, M., Kohnert, A., Wassermann, A.: Optimal linear codes from matrix groups. *IEEE Transactions on Information Theory* 51(12), 4247–4251 (2005)
6. Braun, M.: Some new designs over finite fields. *Bayreuther Math. Schr.* 74, 58–68 (2005)
7. Braun, M., Kerber, A., Laue, R.: Systematic construction of  $q$ -analogs of  $t$ - $(v, k, \lambda)$ -designs. *Des. Codes Cryptography* 34(1), 55–70 (2005)

8. Braun, M., Kohnert, A., Wassermann, A.: Construction of  $(n, r)$ -arcs in  $PG(2, q)$ . *Innov. Incidence Geom.* 1, 133–141 (2005)
9. Drudge, K.: On the orbits of Singer groups and their subgroups. *Electronic Journal Comb.* 9(1), 10 p. (2002)
10. Etzion, T., Silberstein, N.: Construction of error-correcting codes for random network coding (submitted, 2008) (in arXiv 0805.3528)
11. Etzion, T., Vardy, A.: Error-Correcting codes in projective space. In: *ISIT Proceedings*, 5 p. (2008)
12. Gadouleau, M., Yan, Z.: Constant-rank codes and their connection to constant-dimension codes (submitted, 2008) (in arXiv 0803.2262)
13. Kötter, R., Kschischang, F.: Coding for errors and erasures in random network coding. *IEEE Transactions on Information Theory* 54(8), 3579–3391 (2008)
14. Kramer, E.S., Mesner, D.M.:  $t$ -designs on hypergraphs. *Discrete Math.* 15, 263–296 (1976)
15. Maruta, T., Shinohara, M., Takenaka, M.: Constructing linear codes from some orbits of projectivities. *Discrete Math.* 308(5-6), 832–841 (2008)
16. Niskanen, S., Östergård, P.R.J.: Cliquer user's guide, version 1.0. Technical Report T48, Communications Laboratory, Helsinki University of Technology, Espoo, Finland (2003)
17. Schwartz, M., Etzion, T.: Codes and anticodes in the Grassman graph. *J. Comb. Theory, Ser. A* 97(1), 27–42 (2002)
18. Silberstein, N.: Coding theory in projective space. Ph.D. proposal (2008) (in arXiv 0805.3528)
19. Thomas, S.: Designs over finite fields. *Geom. Dedicata* 24, 237–242 (1987)
20. Thomas, S.: Designs and partial geometries over finite fields. *Geom. Dedicata* 63(3), 247–253 (1996)
21. Tonchev, V.D.: Quantum codes from caps. *Discrete Math.* (to appear, 2008)
22. Wassermann, A.: Lattice point enumeration and applications. *Bayreuther Math. Schr.* 73, 1–114 (2006)
23. Xia, S.-T., Fu, F.-W.: Johnson type bounds on constant dimension codes (submitted, 2007) (in arXiv 0709.1074)
24. Zwanzger, J.: A heuristic algorithm for the construction of good linear codes. *IEEE Transactions on Information Theory* 54(5), 2388–2392 (2008)

# Invited Talk: Embedding Classical into Quantum Computation

Richard Jozsa

Department of Computer Science,  
University of Bristol,  
Merchant Venturers Building,  
Bristol BS8 1UB U.K.

**Abstract.** We describe a simple formalism for generating classes of quantum circuits that are classically efficiently simulatable and show that the efficient simulation of Clifford circuits (Gottesman-Knill theorem) and of matchgate circuits (Valiant's theorem) appear as two special cases. Viewing these simulatable classes as subsets of the space of all quantum computations, we may consider minimal extensions that suffice to regain full quantum computational power, which provides an approach to exploring the efficacy of quantum over classical computation.

## 1 Introduction

The characterisation of the possibilities and limitations of quantum computational power is one of the most interesting issues in quantum information science. All of the early and best known quantum algorithms [1] that exhibit an exponential time speed-up over any known classical algorithm for the task, utilize properties of the quantum Fourier transform modulo  $N$ . One may then develop generalisations of these insights, studying Fourier transforms over further abelian and non-abelian groups and invent associated computational tasks such as the hidden subgroup problem and various kinds of hidden shift problems. Around the years of 1997 and 1998 Thomas Beth, with memorable characteristic exuberance, was one of the earliest workers in the subject to recognise the potential possibilities of the abstract formalism of Fourier transforms for novel quantum algorithms, and take up this line of development which has now become an important cornerstone in our understanding.

Despite this seminal development it is probably fair to say that apart from the Fourier transform formalism, no other similarly fruitful quantum algorithmic primitive for exponential speed-up has been identified. This motivates a need for alternative approaches to exploring the efficacy of quantum vs. classical algorithms. One interesting such approach is the identification and study of classes of quantum computations that are classically efficiently simulatable i.e. processes which although quantum, do *not* offer computational benefit. Indeed the relation of classical to quantum computation that emerges is intriguingly rich

and multi-faceted – (sub-) classical computation can be embedded into quantum computation in many inequivalent ways. Given any such class of simulatable quantum computations we may ask: what kind of added (minimal) ingredient suffices to restore full quantum computational power? In a sense, any such ingredient may be viewed as an “essence” of quantum computational power, albeit *relative* to a given substrate of simulatable processes. In this talk we will outline a formalism for providing simulatable classes of quantum circuits and discuss two examples – the Gottesman-Knill theorem for Clifford circuits and Valiant’s theorem for simulation of matchgate circuits. These examples will show that the added ingredient above can be strikingly trivial, especially if thought of as a competitor to the oft-quoted blanket attribution of quantum computational power to the enigmatic phenomenon of entanglement.

## 2 Classically Simulatable Quantum Computations

We focus on comparing and contrasting two theorems of classical simulation *viz.* the Gottesman-Knill theorem for Clifford circuits [1,9] and Valiant’s theorem [4,2] for simulation of matchgate circuits. At first sight these appear to be very different in their content and provenance but we will outline a proof method that reveals a formal similarity between the two results.

The Gottesman-Knill (GK) theorem arose out of the development of the so-called stabiliser formalism for the theory of quantum error correction [1]. Let  $H$  denote the 1-qubit Hadamard gate,  $P$  the 1-qubit phase gate  $P = \text{diag}(1, i)$  and  $CZ$  the 2-qubit controlled- $Z$  gate  $CZ = \text{diag}(1, 1, 1, -1)$ . These gates and arbitrary circuits of them on  $n$  qubits are called *Clifford* operations on  $n$  qubits. Our adopted version (slightly modified from the original, c.f. also [3]) of the GK theorem is the following.

**Theorem 1.** *Consider any uniform (hence poly sized) quantum circuit family comprising the gates  $H, P$  and  $CZ$  (i.e. a Clifford circuit) such that:*

- (i) *the input state is any product state;*
- (ii) *the output is a final  $Z$  measurement on any single qubit line.*

*Then the output may be classically efficiently simulated.*

More formally our notion of efficient classical simulation is the following: given a description of the circuit on  $n$  qubit lines, the output probabilities may be classically computed to  $k$  digits in  $\text{poly}(n, k)$  time.

Next we introduce the notion of “matchgate” and Valiant’s classical simulation theorem [4], which arose originally from considerations of counting perfect matchings in graphs.

A matchgate [4,2] is defined to be any 2-qubit gate  $G(A, B)$  of the form (in the computational basis):

$$G(A, B) = \begin{pmatrix} p & 0 & 0 & q \\ 0 & w & x & 0 \\ 0 & y & z & 0 \\ r & 0 & 0 & s \end{pmatrix} \quad A = \begin{pmatrix} p & q \\ r & s \end{pmatrix} \quad B = \begin{pmatrix} w & x \\ y & z \end{pmatrix} \quad (1)$$

where  $A$  and  $B$  are both in  $SU(2)$  or both in  $U(2)$  with the *same determinant*. Thus the action of  $G(A, B)$  amounts to  $A$  acting in the even parity subspace (spanned by  $|00\rangle$  and  $|11\rangle$ ) and  $B$  acting in the odd parity subspace (spanned by  $|01\rangle$  and  $|10\rangle$ ).

Our version of Valiant's theorem (again slightly different from the original version) is the following.

**Theorem 2.** *Consider any uniform (hence poly-sized) quantum circuit family comprising only  $G(A, B)$  gates such that:*

(i) *the  $G(A, B)$  gates act on nearest neighbour ( $n.n.$ ) lines only;*

(ii) *the input state is any product state;*

(iii) *the output is a final measurement in the computational basis on any single line.*

*Then the output may be classically efficiently simulated.*

Let us now return to the GK theorem and its proof ingredients. The essential property of the class of gates used, i.e. Clifford gates, is the following [9]: if  $C$  is any  $n$ -qubit Clifford operation and  $P_1 \otimes \dots \otimes P_n$  is any product of Pauli matrices (i.e.  $P_i = I, X, Y$  or  $Z$  for each  $i$ ) then the conjugate  $C^\dagger(P_1 \otimes \dots \otimes P_n)C = P'_1 \otimes \dots \otimes P'_n$  is again a product of Pauli operations. Stated more formally, if  $\mathcal{P}_n$  is the group generated by all such Pauli products on  $n$  qubits then the  $n$ -qubit Clifford group is the normaliser of  $\mathcal{P}_n$  in the unitary group  $U(2^n)$ .

A standard proof (c.f. [1]) of the GK theorem (with a computational basis input) proceeds by updating the stabiliser description of the state through the course of the computation and we get a description of the final state in addition to the output probabilities. We adopt here a different approach [3]. Suppose (wlog) that the final measurement is on the first line, having outputs 0,1 with probabilities  $p_0, p_1$  respectively. Then the difference  $p_0 - p_1$  is given by the expectation value of  $Z_1 = Z \otimes I \otimes \dots \otimes I$  in the final state  $C|\psi_0\rangle$ :

$$p_0 - p_1 = \langle \psi_0 | C^\dagger Z_1 C | \psi_0 \rangle \quad (2)$$

This computation suffices to simulate the output (as also  $p_0 + p_1 = 1$ ). Now  $Z_1$  is clearly a product of Pauli operations so  $C^\dagger Z_1 C$  also has the product form  $P_1 \otimes \dots \otimes P_n$  for Pauli operations  $P_i$  (whose identity can be determined in linear time by an update rule for successive conjugations by the elementary gates in the circuit). Hence if  $|\psi_0\rangle = |a_1\rangle \dots |a_n\rangle$  is any product state we get

$$p_0 - p_1 = \prod_{k=1}^n \langle a_k | P_k | a_k \rangle \quad (3)$$

which can clearly be calculated in time  $O(n)$  (as a product of  $n$  terms of fixed size) giving an efficient (linear time) simulation of the Clifford circuit.

The essential ingredients of the above proof are the following.

**(SIM1):** we have a set  $\mathcal{S}_n$  of  $n$ -qubit operations such that  $\langle \psi_0 | S | \psi_0 \rangle$  can be computed in  $\text{poly}(n)$  time for any  $S \in \mathcal{S}_n$  and any allowed input state  $|\psi_0\rangle$ ;

(For the GK theorem  $\mathcal{S}_n$  is the  $n$ -qubit Pauli group  $\mathcal{P}_n$ .)

**(SIM2):** we have a class  $\mathcal{K}_n$  of unitary operations such that  $K^\dagger SK \in \mathcal{S}_n$  for all  $S \in \mathcal{S}_n$  and  $K \in \mathcal{K}_n$ .

(For the GK theorem  $\mathcal{K}_n$  is the Clifford group  $\mathcal{C}_n$ .)

Then if  $Z_1$  is in  $\mathcal{S}_n$  for all  $n$  (or can be expressed in suitably simple terms using elements of  $\mathcal{S}_n$ , c.f. later) then it follows (just as in the above outlined proof) that circuits of gates from  $\mathcal{K}_n$ , with input state  $|\psi_0\rangle$  and output measurement of  $Z$  on the first line, can be classically efficiently simulated.

Note that this simulation result, resting on (SIM1) and (SIM2) does not actually require any special group (or other algebraic) structure on  $\mathcal{S}_n$  or  $\mathcal{K}_n$ . For example, the fact that  $\mathcal{P}_n$  is a *group* is not needed at all in our proof of the GK theorem in contrast to the usual proof resting on the stabiliser formalism, depending heavily on the subgroup structure of  $\mathcal{P}_n$ .

Turning now to matchgates we will show that Valiant's theorem can be understood as just another example of the above formalism with a suitably clever choice of  $\mathcal{S}_n$  and  $\mathcal{K}_n$ . For  $n$  qubits we introduce the  $2n$  Pauli product operators (omitting tensor product symbols  $\otimes$  throughout):

$$\begin{aligned} c_1 &= X I \dots I & c_3 &= Z X I \dots I & \dots & c_{2k-1} &= Z \dots Z X I \dots I \\ c_2 &= Y I \dots I & c_4 &= Z Y I \dots I & \dots & c_{2k} &= Z \dots Z Y I \dots I \end{aligned} \quad (4)$$

where  $X$  and  $Y$  are in the  $k^{\text{th}}$  slot for  $c_{2k-1}$  and  $c_{2k}$ , and  $k$  ranges from 1 to  $n$ . For  $\mathcal{S}_n$  we take the *linear span* of  $c_1, \dots, c_{2n}$  which is a  $2n$ -dimensional vector space (in contrast to the group  $\mathcal{P}_n$ ). Since each  $c_j$  is a product operator and a general vector  $v \in \mathcal{S}_n$  is a linear combination of only  $2n$  of them, it is clear that  $\langle \psi_0 | v | \psi_0 \rangle$  is poly( $n$ )-time computable if  $|\psi_0\rangle$  is a product state i.e. (SIM1) is satisfied.

Next we can verify by straightforward direct calculation that if  $U$  is any n.n.  $G(A, B)$  gate then  $U^\dagger c_j U \in \mathcal{S}_n$  for all  $j$  so  $U^\dagger v U \in \mathcal{S}_n$  for any  $v \in \mathcal{S}_n$  i.e. property (SIM2) is satisfied. More explicitly note that if  $U$  is a n.n.  $G(A, B)$  gate, it applies to two *consecutive* qubit lines so (from eq. (4)) the part of  $c_j$  that it “sees” can only be one of

$$\alpha_1 = ZZ \quad \alpha_2 = ZX \quad \alpha_3 = ZY \quad \alpha_4 = XI \quad \alpha_5 = YI \quad \text{or} \quad \alpha_6 = II. \quad (5)$$

Then a straightforward calculation with 4 by 4 matrices shows that for each  $i$ ,  $G(A, B)^\dagger \alpha_i G(A, B)$  always returns a linear combination of allowable  $\alpha_i$ 's and property (SIM2) follows immediately.

It is instructive to note that if we attempt to apply a  $G(A, B)$  gate on *not* nearest-neighbour qubit lines then in addition to the six terms in eq. (5) we can get a *further* possibility, namely  $\alpha_7 = ZI$  on the chosen two lines. But now we can check that  $G(A, B)^\dagger \alpha_7 G(A, B)$  does *not* generally lie in the span of the allowed Pauli products at those lines, and property (SIM2) is violated. This give a way of understanding the curious n.n. requirement for  $G(A, B)$  actions in theorem 2, which has no analogue in the GK theorem (as  $\mathcal{P}_n$  is defined by a uniformly local product requirement).

With properties (SIM1) and (SIM2) we can say that if  $M$  is the total operation of any n.n. matchgate circuit on  $n$  lines then  $\langle \psi_0 | M^\dagger D M | \psi_0 \rangle$  is poly( $n$ )-time

computable for any  $D \in \mathcal{S}_n$ . To complete our simulation theorem we want to set  $D = Z_k = I \dots I Z I \dots I$  (i.e.  $Z$  on the  $k^{\text{th}}$  line) to obtain  $p_0 - p_1$  for a measurement on the  $k^{\text{th}}$  line. In the GK theorem with  $\mathcal{S}_n = \mathcal{P}_n$  we had  $Z_k \in \mathcal{P}_n$  directly. In the present case we do not have  $Z_k \in \mathcal{S}_n$  but looking at eq. (4) we see that  $Z_1 = -ic_1c_2$  and generally  $Z_k = -ic_{2k-1}c_{2k}$ . Then, for example,

$$M^\dagger Z_1 M = -iM^\dagger c_1 c_2 M = -i(M^\dagger c_1 M)(M^\dagger c_2 M) \quad (6)$$

and each bracket in the last expression is a linear combination of  $c_j$ 's. Thus  $p_0 - p_1 = \langle \psi_0 | M^\dagger Z_1 M | \psi_0 \rangle$  has the form  $-i \sum_{ij} a_i b_j \langle \psi_0 | c_i c_j | \psi_0 \rangle$ . Since the  $c_i$ 's are product operators, so are the  $O(n^2)$  product terms  $c_i c_j$  in the final sum. Hence  $p_0 - p_1$  is again poly( $n$ )-time computable but now we have  $O(n^2)$  terms instead of the previous  $O(n)$  terms in the sum. This completes a proof outline of Valiant's theorem 2.

### 3 Extensions of Simulatable Circuits

We may now view Clifford circuits and matchgate circuits as two “islands” of quantum processes in the space of all quantum computations, that offer no computational time benefit over classical computations. As such, it is interesting to try to characterise their relationship to the whole and one approach is to consider what (minimal) extra ingredient suffices to expand their computational power to regain full universal efficient quantum computation.

In the case of Clifford circuits it is well known (e.g. see [1]) that the inclusion of the phase gate  $\sqrt{P} = \text{diag}(1, e^{i\pi/4})$  suffices, and more generally, (using a result of Shi [10], noting that  $CNOT$  is a Clifford operation), the inclusion of essentially any single extra non-trivial 1-qubit gate will suffice.

For the case of matchgate circuits we have the following intriguing result.

**Theorem 3.** *Let  $C_n$  be any uniform family of quantum circuits with output given by a  $Z$  basis measurement on the first line. Then  $C_n$  may be simulated by a circuit of  $G(A, B)$  gates acting on n.n. or next n.n. lines only (i.e. on line pairs at most distance 2 apart) with at most a constant factor increase in the size of the circuit.*

A proof of this theorem may be found in [2] and here we just make a few remarks. Comparing theorems 2 and 3 we see that the gap between classical and full quantum computational power can be bridged by a very modest use of a seemingly innocuous resource viz. the ability of matchgates to act on next n.n. – instead of just n.n. – qubit lines. Equivalently this may be characterised by use of the  $SWAP$  operation (on n.n. lines) in a very constrained context where *ladders* of consecutive  $SWAP$ s (which would allow 2-qubit gates to act on arbitrarily distant lines) are not even allowed. From this perspective, the power of quantum (over classical) computation is attributable to the mere inclusion of such isolated single  $SWAP$  gates. The result becomes perhaps even more striking if we note that  $SWAP$  itself is very close to being expressible in the allowed  $G(A, B)$

form. Indeed  $SWAP = G(I, X)$  and fails only through a mere minus sign in  $\det X = -\det I$ . Thus if we drop the  $\det A = \det B$  condition in eq. (1), then the resulting  $G(A, B)$  gates acting on n.n. lines become efficiently universal for quantum computation.

Is it conceivable that the passage from n.n. to next-n.n. use of  $G(A, B)$  gates may be achieved while maintaining classical simulatability? We may argue on formal complexity theoretic grounds that this is highly implausible. Indeed it is shown in [2] that the classical complexity classes NP and PP (cf. [7]) would then become classically poly-time decidable i.e. we would get  $P=NP=PP$  (as well as  $P=BQP$ ). Thus an extra supra-classical computational power *must* be associated to the single distance extension of the range of n.n. 2-qubit  $G(A, B)$  gates in general matchgate circuits, if these classical computational complexity classes are to be unequal.

## 4 Concluding Remarks

From the viewpoint of (SIM1) and (SIM2) we see a formal similarity between the GK theorem and Valiant's theorem although these results arose historically from very different considerations. This suggests that we might be able to construct further interesting classes of classically simulatable circuits by simply taking other choices of  $\mathcal{S}_n$  and identifying a suitable associated  $\mathcal{K}_n$ . However "interesting" pairs  $(\mathcal{S}_n, \mathcal{K}_n)$  appear to be difficult to invent – the known examples arising as outcomes of some prior elaborate underlying mathematical structures. In the GK case we have the identification of the Clifford group via a lengthy argument with group theoretic ingredients (see e.g. appendix in [11]) applied to the Pauli group  $\mathcal{P}_n$  which is a well known structure in the subject.

However in the case of Valiant's theorem, how might we initially come upon this result, and guess the choice for  $\mathcal{S}_n$  that we used (i.e. eq. (4) and its linear span)!? Actually the operators in eq. (4) are well known in physics – they comprise the so-called Jordan-Wigner representation [8] that appears in the theory of non-interacting fermions. The connection between Valiant's theorem and simulation of free fermions was recognised by Knill [5] and Terhal and DiVincenzo [6] and our proof of Valiant's theorem above is a re-writing of this connection. A more formal mathematical treatment (albeit without reference to fermions) based on abstract properties of the mathematical structure of Clifford algebras is given in [2] which also clarifies the appearance of matchgates as normalisers of the linear part of the Clifford algebra, leading to property (SIM2). We will not elaborate here on these further ingredients (detailed in [2]) except to point out that again here, we have a significant underlying theory leading to the choice of  $\mathcal{S}_n$  and the identification of its associated normalisers  $\mathcal{K}_n$ . Perhaps an intuitive signal feature of such an underlying theory is some construction that could potentially produce an exponentially large structure but surprisingly remains only polynomially complex. In the case of the Pauli group  $\mathcal{P}_n$ , conjugation by arbitrary  $V \in U(2^n)$  can generate general  $n$ -qubit matrices for which the calculation of the expectation value in eq. (2) becomes exponentially inefficient. But

the special case of  $V$  being Clifford guarantees a polynomial simplicity via the preserved product structure. In the case of the  $c_i$ 's of eq. (4), conjugation by an arbitrary  $V \in U(2^n)$  leads to a general element of the full Clifford algebra generated by the  $c_i$ 's [2] – a space of exponential dimension  $2^{2n}$  – but again the special case of n.n. matchgates (associated to a theory of quadratic hamiltonians [2]) guarantees that the conjugates remain in the polynomially small subspace of linear elements of the full Clifford algebra. It is an interesting open problem to exhibit further examples of such simplifications and of our formalism (SIM1), (SIM2), that may already exist within the literature of the theory of some yet more general kind of algebraic structure.

**Acknowledgements.** This work was supported in parts by the EC networks QICS and QAP and by EPSRC QIP-IRC. The author also acknowledges Akimasa Miyake for the collaborative work [2] which is related closely to the discussion in this paper.

## References

1. Nielsen, M., Chuang, I.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2000)
2. Jozsa, R., Miyake, A.: Appearing as Proc. R. Soc (Lond.) A 464, 3089–3106 (2008); arXiv:quant-ph/0804.4050
3. Clark, S., Jozsa, R., Linden, N.: Quant. Inf. Comp. 8, 106–126 (2008)
4. Valiant, L.: SIAM J. Computing 31(4), 1229 (2002)
5. Knill, E.: (2001); arXiv:quant-ph/0108033
6. Terhal, B., DiVincenzo, D.: Phys. Rev. A 65, 032325 (2002)
7. Papadimitriou, C.: Computational Complexity. Addison-Wesley, Reading (1994)
8. Jordan, P., Wigner, E.: Zeitschrift für Physik. 47, 631 (1928)
9. Gottesman, D.: Stabilizer Codes and Quantum Error Correction, PhD thesis, California Institute of Technology, Pasadena, CA (1997)
10. Shi, Y.: Quant. Inf. Comp. 3, 84–92 (2003)
11. Clark, S.: J. Phys. A: Math. Gen. 39, 2701–2721 (2006)

# A Criterion for Attaining the Welch Bounds with Applications for Mutually Unbiased Bases

Aleksandrs Belovs<sup>1,\*</sup> and Juris Smotrovs<sup>2,\*\*</sup>

<sup>1</sup> Department of Combinatorics and Optimization, and Institute for Quantum Computing, University of Waterloo, Waterloo, Ontario, Canada, N2L 3G1

<sup>2</sup> Department of Computer Science, University of Latvia,  
 Raiņa bulvāris 19, Rīga, Latvia

**Abstract.** The paper gives a short introduction to mutually unbiased bases and the Welch bounds and demonstrates that the latter is a good technical tool to explore the former. In particular, a criterion for a system of vectors to satisfy the Welch bounds with equality is given and applied for the case of MUBs. This yields a necessary and sufficient condition on a set of orthonormal bases to form a complete system of MUBs.

This condition takes an especially elegant form in the case of homogeneous systems of MUBs. We express some known constructions of MUBs in this form. Also it is shown how recently obtained results binding MUBs and some combinatorial structures (such as perfect nonlinear functions and relative difference sets) naturally follow from this criterion.

Some directions for proving non-existence results are sketched as well.

## 1 Mutually Unbiased Bases

The current research originated in the problem of constructing a complete set of mutually unbiased bases and is inspired mostly by [22].

A set of *mutually unbiased bases* (MUBs) in the Hilbert space  $\mathbb{C}^n$  is defined as a set of orthonormal bases  $\{B_0, B_1, \dots, B_r\}$  of the space such that the absolute value of a scalar product  $|\langle x|y\rangle|$  is equal to  $\frac{1}{\sqrt{n}}$  for any two vectors  $x \in B_i$ ,  $y \in B_j$  with  $i \neq j$ . For the sake of brevity we will further call the absolute value of a scalar product of two vectors as the *angle* between these vectors. We will often group vectors of a basis into a matrix and say that two unitary matrices are mutually unbiased iff the bases obtained from their columns are. Bases with such properties were first observed by Schwinger in [30]. The name of mutually unbiased bases is due to Fields and Wootters [34].

The applications of MUBs include quantum state determination [21,34], quantum cryptography (the protocol BB84 due to Bennet and Brassard [5] is a classical example of such a usage), the Mean King's problem [1] and Wigner functions [35]. A good source of an up-to-date information on MUBs can be found on [13].

Clearly, if  $n = 1$  then any number of unit vectors (in fact, scalars) gives a set of MUBs. This result does not seem very useful, so we will further assume

---

\* Research supported in part by MITACS and Province of Ontario.

\*\* Supported by the European Social Fund and by the University of Latvia research project No Y2-ZP14-100.

the dimension of the space  $n$  is at least 2. In this case it can be proved that the number of bases in any set of MUBs in  $\mathbb{C}^n$  doesn't exceed  $n + 1$  (see Theorem 3 later in the text). A set of bases that achieves this bound is called a *complete set of MUBs*. An interesting question is whether such a set exists for any given dimension  $n$ . The answer is positive if  $n$  is a prime power [21,34]. The corresponding constructions are listed in section 9 of this paper. In all other cases (even for  $n = 6$ ) the question is still open, despite a considerable effort spent on solving this problem (see, e.g., [4]).

The search for complete systems of MUBs is complicated because of the number of bases we should find and because of the non-obviousness of the value of the angle  $\frac{1}{\sqrt{n}}$ . Using the Welch bounds (described in the next section) we give a sufficient and necessary condition that uses solely orthogonality of vectors. Clearly, it is a much more studied and intuitive relation.

This is not the first attempt to substitute the angle  $\frac{1}{\sqrt{n}}$  by zero. An alternative approach appears in the classical paper [34]:

**Proposition 1.** *Consider the operation that maps a state  $|x\rangle \in \mathbb{C}^n$  to the matrix  $Y_x = |x\rangle\langle x| - I/n$ . Then  $|\langle x|y\rangle| = \frac{1}{\sqrt{n}}$  if and only if matrices  $Y_x$  and  $Y_y$  are orthogonal with respect to the trace inner product:  $\text{Tr}(Y_x^\dagger Y_y) = 0$ .*

In particular, applications of MUBs in quantum state tomography are based on this observation.

Our approach is slightly different. Using the collection of  $n + 1$  orthonormal bases in  $\mathbb{C}^n$ , pretending to be mutually unbiased, we build  $n$  flat (with all entries having the same absolute value) vectors, each in  $\mathbb{C}^{n^2}$ . Next, from each pair of these vectors we obtain a new vector from the same space. We prove that the bases of the original collection are MUBs if and only if the latter vectors are pairwise orthogonal. It is not a problem to find  $\binom{n}{2}$  orthogonal flat vectors in  $\mathbb{C}^{n^2}$ , but, in general, they won't be decomposable back to pairs.

Moreover, if we restrict our attention to homogeneous systems of MUBs (see Section 6 for the definition), it is possible to reduce the criterion to only two matrices from  $\mathbb{C}^n$  and orthogonality conditions obtained in a similar fashion. In order to show the usability of our result we show how it sheds light on the known constructions of complete sets of MUBs. In particular, we give a bit easier proofs that these constructions do result in complete sets of MUBs.

We also show how this approach naturally leads to some applications of combinatorial structures to MUBs that were obtained recently. In particular, we extend the correspondence between planar functions and splitting semiregular relative difference sets to the case of non-splitting ones.

## 2 Welch Bounds and Crosscorrelation

Welch bounds are the inequalities from the following theorem:

**Theorem 2.** *For any finite sequence  $\{x_i\}$  of vectors in Hilbert space  $\mathbb{C}^n$  and any integer  $k \geq 1$  the following inequality holds:*

$$\binom{n+k-1}{k} \sum_{i,j} |\langle x_i | x_j \rangle|^{2k} \geq \left( \sum_i \langle x_i | x_i \rangle^k \right)^2. \quad (1)$$

The proof will be given in Section 4, but for now let us note that these inequalities were first derived (in the case of all vectors having the same norm) by Welch in [33]. It is worth to become acquainted with his motivation.

In order to do this we should define sequences with low correlation. For a systematic treatment of the topic see [18]. Let  $u$  and  $v$  be complex periodic sequences of equal period  $n$ . Usually the sequences are defined as  $u_i = \omega_q^{a_i}$  with  $a_i$  from  $\mathbb{Z}_q$ . ( $\omega_q$  is a primitive  $q$ -th power root of unity:  $\omega_q = e^{\frac{2\pi i}{q}}$ ,  $\mathbb{Z}_q$  is the ring of integers modulo  $q$ ). The binary case (with  $q = 2$ ) is the most common. The (*periodic*) correlation of  $u$  and  $v$  is defined as (where  $L$  stands for the left cyclic shift function)

$$\theta_{u,v}(\tau) = \langle L^\tau(u) | v \rangle = \sum_{i=1}^n \overline{u_i} v_{i+\tau}.$$

The correlation of a sequence with itself is called its *autocorrelation*  $\theta_u(\tau) = \langle L^\tau(u) | u \rangle$ . The correlation of two shift-distinct sequences is usually called *cross-correlation*.

Informally, the correlation of binary sequences characterizes the number of places two sequences coincide minus the number of places they differ. For random sequences magnitude of this value is small, so it can be used as a measure of the pseudorandomness of a sequence. The correlation is called *ideal* if it is as small as possible (0 or  $\pm 1$ ). It is considered low, if it is  $O(\sqrt{n})$  (an expected value for random sequences). For example, *m-sequences* (the maximal length sequences generated by a linear feedback shift register (LFSR)) have ideal autocorrelation, since for them  $\theta(\tau) = -1$  for any  $\tau \not\equiv 0 \pmod{n}$ . This, among other properties, explains why they are used in cryptography (as a main building block of nearly every stream cipher) and electronic engineering (e.g., in radars).

Families of sequences with low crosscorrelation are also well-studied. A nice property of these sequences is that they can be transmitted through the same channel simultaneously without mutual disturbance. By the time Welch was writing his paper there were some good families of sequences with low auto- and cross-correlation and he got interested in obtaining upper bounds on the number of sequences in a family.

For example, one classical family of sequences was proposed by Gold in [15]. For any integer  $k$  he constructed a family of  $2^k + 1$  binary sequences of period  $2^k - 1$  and correlation between any two of them takes only three possible values:  $-1$ ,  $-(2^{(k+1)/2} + 1)$  and  $2^{(k+1)/2} - 1$ .

Similarity of this family and a complete family of MUBs is apparent. Both are built of vectors from  $\mathbb{C}^n$ , vectors are joined in blocks of size  $n$ , the number of blocks is approximately the same and the ratios of possible inner products and norms of vectors also almost agree. So, an attempt to apply Welch bounds to the problem of MUBs seems quite reasonable.

Even more, it turns out that Alltop in his work [2] of 1980 (i.e., one year before the work [21] of Ivanović) for any prime  $p \geq 5$  gave a set of  $p$  sequences with period  $p$  and elements with absolute value  $\frac{1}{\sqrt{p}}$ , such that the crosscorrelation is given by

$$|\theta_{uv}(\tau)| = \begin{cases} 1, & u = v \text{ and } \tau = 0; \\ 0, & u \neq v \text{ and } \tau = 0; \\ \frac{1}{\sqrt{p}}, & \tau \neq 0. \end{cases}$$

Clearly, these sequences with different shifts and the standard basis give a complete set of MUBs in  $\mathbb{C}^p$ . This result was generalized to prime power dimensions in [23].

### 3 Link between MUBs and the Welch Bounds

In our first application of the Welch bounds to MUBs we can apply the original approach of Welch in the new settings. It is easy to check that a union of orthonormal bases satisfy the Welch bound for  $k = 1$  (it can be done either directly using (1) or using Theorem 5 further in the text). So, we should use  $k = 2$ .

**Theorem 3.** *If  $n \geq 2$  then the maximal number of mutually unbiased bases in  $\mathbb{C}^n$  does not exceed  $n + 1$ .*

*Proof.* Suppose we have a system of  $n + 2$  MUBs. Join all vectors of the system into one big sequence  $\{x_i\}$  of size  $n(n + 2)$ . Let us fix  $k = 2$  and calculate the left hand side of (1). We have  $n(n + 2)$  vectors, each giving the scalar product 1 with itself and  $n(n + 1)$  scalar products of absolute value  $\frac{1}{\sqrt{n}}$  with vectors from other bases. Summing up, we have:

$$\begin{aligned} \binom{n+1}{2} \sum_{i,j} |\langle x_i | x_j \rangle|^4 &= \frac{n(n+1)}{2} \left[ n(n+2) \left( 1 + n(n+1) \cdot \frac{1}{n^2} \right) \right] \\ &= \frac{n(n+1)(n+2)(2n+1)}{2}. \end{aligned}$$

For the right hand side we have:

$$\left( \sum_i \langle x_i | x_i \rangle^2 \right)^2 = n^2(n+2)^2 > \frac{n(n+1)(n+2)(2n+1)}{2},$$

in a contradiction with the Welch bound for  $k = 2$ . □

Originally it was proved in [34] using the result of Proposition 1.

If we reduce the number of MUBs from  $n + 2$  to  $n + 1$  we don't get an apparent contradiction. However, even in this case the Welch bounds prove themselves to be useful.

**Theorem 4.** *Let  $\{B_i\}$  be a set of  $n + 1$  orthonormal bases in an  $n$ -dimensional Hilbert space and  $X$  be the union of these bases (that is the sequence of vectors, each of them appearing in the sequence the same number of times it appears in the bases). Then  $X$  satisfies the Welch bound for  $k = 2$  with equality if and only if  $\{B_i\}$  form a complete system of MUBs.*

*Proof.* If  $\{B_i\}$  is a complete system of MUBs and  $X = \{x_i\}$  is the union of its bases, then calculations similar to ones in the proof of Theorem 3 show

$$\binom{n+1}{2} \sum_{i,j} |\langle x_i | x_j \rangle|^4 = \frac{n(n+1)}{2} \left[ n(n+1) \left( 1 + n^2 \cdot \frac{1}{n^2} \right) \right] = n^2(n+1)^2$$

and

$$\left( \sum_i \langle x_i | x_i \rangle^2 \right)^2 = n^2(n+1)^2.$$

And vice versa, suppose  $X$ , being a union of orthonormal bases, attains the Welch bound for  $k = 2$ . Then,  $|\langle x | x \rangle|^4 = 1$  for each  $x$  in  $X$ ,  $|\langle x | y \rangle|^4 = 0$  for two different vectors of the same basis, and by the inequality between square and arithmetic means we get:

$$\sum_{x \in B_i} |\langle x | y \rangle|^4 \geq \frac{1}{n} \left( \sum_{x \in B_i} |\langle x | y \rangle|^2 \right)^2 = \frac{1}{n}.$$

for any vector  $y$  of unit length. To attain the Welch bound, this inequality must actually be an equality, which is achieved only if  $|\langle x | y \rangle|^2$  has the same value for all vectors  $x$  from  $B_i$ . This means that bases  $\{B_i\}$  form a complete system of MUBs.  $\square$

Systems of vectors attaining the Welch bounds have been investigated before. A system of unit-norm vectors from  $\mathbb{C}^n$  attaining the Welch bounds for all  $k \leq t$  is called a *complex projective  $t$ -design*. This is a Chebyshev-type averaging set on the  $n$ -dimensional complex unit sphere  $\mathbb{C}S^{n-1}$ , in the sense that the integral of every polynomial of degree  $\leq t$  is equal to the average of its values on the vectors from the  $t$ -design. See [22] for more details.

Analysis of links between complex projective 2-designs and MUBs was initiated in Zauner's dissertation [36] and was continued in the work by Klappenecker and Rötteler [22]. In particular, the 'if' part of Theorem 4 is due to them. The 'only if' part seems first to appear later, in [28].

We give a criterion for attaining the Welch bounds in the next section.

## 4 Criterion for Attaining the Welch Bounds

Let us at first define the Hadamard product of two matrices. Let  $A = (a_{ij})$  and  $B = (b_{ij})$  be two matrices of equal sizes. The *Hadamard product* (see, for example, chapter 7 of [19]) is the matrix of the same size (denoted by  $A \circ B$ )

with its  $(i, j)$ -entry equal to  $a_{ij}b_{ij}$ . In other words, multiplication is performed component-wise. The  $k$ -th Hadamard power of the matrix  $A$  is again the matrix of the same size (denoted by  $A^{(k)}$ ) with its  $(i, j)$ -entry equal to  $a_{ij}^k$ .

Additionally, we shall use notation  $A^\dagger$  for the adjoint matrix (complex conjugated and transposed) and the term *self-adjoint* for matrices  $A$  satisfying  $A^\dagger = A$  (also called Hermitian).

We will at first give a proof of the Welch bounds and then extract the equality criterion from the proof.

*Proof (of Theorem 2).* Let us construct the Gram matrix  $G = (a_{ij})$  with  $a_{ij} = \langle x_i | x_j \rangle$  and consider its  $k$ -th Hadamard power  $G^{(k)}$  (with  $k$  being a positive integer). The square of its Euclidean norm (see chapter 5 of [19], also known as Frobenius norm and Schur norm) is defined by

$$\left( \|G^{(k)}\|_E \right)^2 = \sum_{i,j} |\langle x_i | x_j \rangle|^{2k} = \tag{2}$$

Unitary operators, applied both from the left and the right, do not change the Euclidean norm of a matrix. Any self-adjoint matrix can be transformed into a diagonal matrix with real entries (its eigenvalues) on the diagonal by a unitary transformation, and  $G^{(k)}$  is a self-adjoint matrix, hence

$$= \sum_{\lambda \in \sigma(G^{(k)})} \lambda^2 \geq \tag{3}$$

here  $\sigma$  is the spectrum (the multiset of the eigenvalues of a matrix). By the inequality between square and arithmetic means, we have (let us remind that the rank of a self-adjoint matrix is equal to the number of its non-zero eigenvalues):

$$\geq \frac{1}{\text{rank}(G^{(k)})} (\text{Tr } G^{(k)})^2 \geq \frac{1}{\binom{n+k-1}{k}} \left( \sum_i \langle x_i | x_i \rangle^k \right)^2. \tag{4}$$

The last estimation on the rank of  $G^{(k)}$  we will prove later. □

**Theorem 5.** *Let  $B$  be a matrix and  $X \subset \mathbb{C}^n$  be the sequence of its columns. Let  $w_1, w_2, \dots, w_n$  be the rows of the matrix. Then  $X$  attains the Welch bound for a fixed  $k$  if and only if all vectors from*

$$W = \left\{ \sqrt{\binom{k}{k_1, \dots, k_n}} w_1^{(k_1)} \circ w_2^{(k_2)} \circ \dots \circ w_n^{(k_n)} \mid k_i \in \mathbb{N}_0, k_1 + \dots + k_n = k \right\}$$

*are of equal length and pairwise orthogonal.*

In other words, each vector of  $W$  is a Hadamard product of a  $k$ -multiset of rows of  $B$  with a coefficient that is the square root of the multinomial coefficient of the multiset

$$\binom{k}{k_1, \dots, k_n} = \frac{k!}{k_1! k_2! \dots k_n!}.$$

*Proof.* At first, let us note that matrix  $G$  in (2) is equal to  $B^\dagger B$ . So (if each  $w_i$  is treated as a row vector):

$$G = w_1^\dagger w_1 + w_2^\dagger w_2 \cdots + w_n^\dagger w_n.$$

By the formula for a power of a sum, we obtain

$$G^{(k)} = \sum_{k_1 + \cdots + k_n = k} \binom{k}{k_1, \dots, k_n} \left( w_1^{(k_1)} \circ \cdots \circ w_n^{(k_n)} \right)^\dagger \left( w_1^{(k_1)} \circ \cdots \circ w_n^{(k_n)} \right).$$

In other words,  $G^{(k)} = C^\dagger C$ , where the rows of  $C$  are exactly the vectors from  $W$ . This gives the bound on the rank of  $G^{(k)}$  used in (4), because the number of  $k$ -multisets of an  $n$ -set equals  $\binom{n+k-1}{k}$  (see, e.g., Section 1.2 of [29]).

By observing the inequality between (3) and (4), we see that  $X$  satisfies the Welch bound for a fixed  $k$  with equality if and only if  $G^{(k)}$  has  $\binom{n+k-1}{k}$  equal non-zero eigenvalues (all other eigenvalues are automatically zeros due to the rank observations).

It is a well-known fact that for any matrices  $P$  and  $Q$  the set of non-zero eigenvalues of matrices  $PQ$  and  $QP$  are equal whenever these two products are defined (see Section 1.3 of [19]). Hence,  $CC^\dagger$  have  $\binom{n+k-1}{k}$  equal non-zero eigenvalues, and because it is a self-adjoint matrix of the same size it is a scalar multiple of the identity matrix. And the latter is equivalent to the requirement on the set  $W$ .  $\square$

We haven't hitherto seen the pair of Theorems 2 and 5 appearing in such a general form, however all ideas involved in the proof have already appeared in the proofs of other results. As we have already said, Welch was the first who derived the bounds (1) in the case when all vectors have unit norm and  $k$  is arbitrary. It was done in [33]. The variant of Theorem 5, with  $k = 1$  and all vectors of equal length, seems first to appear in [24]. Our proof is a generalization of an elegant proof found in [32]. In the latter paper the Welch bounds are stated in the case of vectors of different length, but it deals with the case of  $k = 1$  only.

## 5 Application of the Criterion to MUBs

At first let us state the following easy consequence of Theorem 5:

**Corollary 6.** *Let  $B$  be a matrix and  $X \subset \mathbb{C}^n$  be the sequence of its columns. Let  $w_1, w_2, \dots, w_n$  be the rows of the matrix. Then  $X$  satisfy the Welch bound for  $k = 2$  with equality if and only if all vectors from  $W = \{w_i^{(2)}\} \cup \{\sqrt{2}w_i \circ w_j \mid 1 \leq i < j \leq n\}$  are of equal length and pairwise orthogonal.*

Suppose we have a complete system of MUBs:  $\{B_0, B_1, \dots, B_n\}$ . We can always represent them in the first basis  $B_0$ , thus we can assume that the first basis is the standard basis (the identity matrix). Then the matrices representing all other bases have all their entries equal by the absolute value to  $\frac{1}{\sqrt{n}}$ .

A matrix with complex entries and with all entries having the same absolute value is called a *flat matrix*. If it is additionally unitary (or a scalar multiple of a unitary), it is called a *complex Hadamard matrix*. It is common to rescale flat matrices in such a way that each its element has absolute value 1. We will usually assume that. In the case of an  $n \times n$  complex Hadamard matrix it is sometimes more convenient to assume each element having absolute value  $\frac{1}{\sqrt{n}}$ , sometimes 1. According to the situation we will use both assumptions, it will be usually clear from the context what is meant.

Complex Hadamard matrix is a generalization of classical Hadamard matrix that satisfies the same requirements, but with all entries real (i.e.,  $\pm 1$ ) (see, for example, Section I.9 of [3]). We will further use term Hadamard matrix or just Hadamard to denote complex Hadamard matrices.

Two Hadamard matrices are called *equivalent* if one can be got from the other using row and column multiplications by a scalar and its permutations. Some classes of equivalent Hadamards are classified. See [31] for more details.

A system of Hadamards such that any two are mutually unbiased is called a *system of mutually unbiased Hadamards* or *MUHs* for short. The following result is obvious

**Proposition 7.** *A complete system of MUBs exists in space  $\mathbb{C}^n$  if and only if there is a system of  $n$  MUHs in the same space.*

A system of  $n$  MUHs in  $\mathbb{C}^n$  is called a *complete system of MUHs*. We will turn to the investigation of complete systems of MUHs in the remaining part of the paper.

Now we are able to prove the following theorem:

**Theorem 8.** *Let  $\{B_i\}$  ( $i = 1, 2, \dots, n$ ) be a set of  $n$  Hadamards in  $\mathbb{C}^n$  and  $B$  be a concatenation of these matrices (i.e. a  $n^2 \times n$ -matrix having as columns all columns appearing in  $\{B_i\}$ ). Then  $\{B_i\}$  form a complete set of MUHs if and only if all vectors from  $W' = \{w_i \circ w_j \mid 1 \leq i \leq j \leq n\}$  are pairwise orthogonal, where  $\{w_i\}$  are the rows of  $B$ .*

*Proof.* Let us denote the  $n \times n$  identity matrix by  $B_0$ . By Theorem 4, we see that the set  $\{B_0, B_1, \dots, B_n\}$  is a complete set of MUBs if and only if the set of columns of all these matrices attains the Welch bound for  $k = 2$ . Now from Corollary 6 it follows that it only remains to show that vectors from  $W$ , as it was defined in Corollary 6, are of equal length and orthogonal if vectors of  $W'$  are orthogonal.

If a vector from  $W$  is multiplied by itself using the Hadamard product, the result has one 1 and all other entries equal to 0 in the part corresponding to  $B_0$ , and all entries in other parts by the absolute value are equal to  $\frac{1}{n}$ . Hence, the length of the vector is  $\sqrt{1 + n^2 \frac{1}{n^2}} = \sqrt{2}$ .

If two distinct vectors are multiplied, the result has only zeroes in the first part, and its length is  $\sqrt{n^2 \frac{1}{n^2}} = 1$ . We see that all vectors from  $W$  have the same length.

Moreover, the part of  $B_0$  contributes zero to the inner product of 2 distinct vectors of  $W$ , hence vectors of  $W$  are orthogonal if and only if the corresponding vectors of  $W'$  are.  $\square$

Let us restate the last theorem. Suppose  $B$  is a flat  $n \times n$ -matrix. Construct the weighted graph  $K(B)$  as follows. Its vertices are all multisets of size 2 from  $\{1, \dots, n\}$ . Semantically a vertex  $\{i, j\}$  represents the Hadamard product of the  $i$ -th and the  $j$ -th row of  $B$ . The weight of an edge is the inner product of the vertices it joins. (Of course, thus defined, the weight depends on the order of the vertices, but let us fix a direction of each edge, say lexicographical). Then Theorem 8 can be restated by saying that Hadamards  $B_1, \dots, B_n$  form a set of MUHs in  $\mathbb{C}^n$  if and only if the sum of weights of each edge in all of  $K(B_1), \dots, K(B_n)$  equals 0. In fact, there is no need to consider edges between vertices that have an element in common, since they will be orthogonal.

It does not seem that this restatement makes the problem much easier comparing to the initial formulation. However, careful examination of the possible configurations of weights that can be achieved in  $K(B)$  may shed some light on the problem. In the next section we consider a special case of systems of MUHs for which Theorem 8 yields a considerable simplification.

## 6 Homogeneous Systems of MUBs

Suppose we have a flat  $n \times n$ -matrix  $A = (a_{i,j})$  and an Hadamard matrix  $H = (h_{i,j})$  of the same dimensions. Consider the following system of Hadamards (it's assumed that each element of  $A$  and  $H$  has absolute value 1)

$$(v_k^{(r)})_\ell = \frac{1}{\sqrt{n}} a_{\ell,r} h_{\ell,k} \quad (5)$$

with  $r$  being a matrix index,  $k$  being a column index and  $\ell$  being a row index ( $r, k, \ell \in \{1, \dots, n\}$ ). In other words, the  $i$ -th matrix is given by  $\text{diag}(v_i)H$ , where  $v_i$  is the  $i$ -th column of  $A$ . We will call such a set of Hadamards (in the case it forms a set of MUHs) a *homogeneous system of MUHs*, or a *homogeneous system of MUBs* if the identity matrix is appended. The name is borrowed from [4].

From Theorem 8 it follows that the system from (5) forms a system of MUHs if and only if

$$\begin{aligned} \langle w_{\ell_1} \circ w_{\ell_2} | w_{\ell_3} \circ w_{\ell_4} \rangle &= \frac{1}{n} \sum_{r,k} \overline{a_{\ell_1,r} h_{\ell_1,k} a_{\ell_2,r} h_{\ell_2,k} a_{\ell_3,r} h_{\ell_3,k} a_{\ell_4,r} h_{\ell_4,k}} = \\ &= \frac{1}{n} \left( \sum_r \overline{a_{\ell_1,r} a_{\ell_2,r} a_{\ell_3,r} a_{\ell_4,r}} \right) \left( \sum_k \overline{h_{\ell_1,k} h_{\ell_2,k} h_{\ell_3,k} h_{\ell_4,k}} \right) = 0 \end{aligned}$$

for all  $\ell_1, \ell_2, \ell_3$  and  $\ell_4$  such that  $\{\ell_1, \ell_2\} \neq \{\ell_3, \ell_4\}$ .

Let us define the  $L$ -graph (denoted  $L(A)$ ) of a flat matrix  $A$  as follows. It is a simple graph with the same set of vertices as  $K(A)$ . Two vertices are adjacent if and only if the corresponding vectors are orthogonal. The previous identity leads to the following observation:

**Proposition 9.** *The homogeneous system given by (5) is a complete system of MUHs if and only if the graphs  $L(A)$  and  $L(H)$  together cover the complete graph.*

If matrices  $A$  and  $H$  satisfy the conditions of Proposition 9 and  $A'$  and  $H'$  are such matrices that  $L(A)$  is a subgraph of  $L(A')$  and the same holds for  $L(H)$  and  $L(H')$ , then  $A'$  and  $H'$  also give rise to a complete system of MUHs via (5). Hence, without loss of generality we may consider only matrices with maximal L-graphs. We will call them L-maximal flat or Hadamard matrices, respectively. What are they? We can say little on the subject at the moment, it is a topic for a future research.

*Problem 10.* Describe L-maximal flat and Hadamard matrices and the corresponding graphs.

An answer to this question would possibly allow a systematization of all complete homogeneous systems of MUBs. Anyway, it is already clear that L-maximal flat matrices cover L-maximal Hadamard matrices (because the latter is a special case of the former).

There is an important class of L-maximal Hadamard matrices. It is a very common example of Hadamard matrices and it is used in all known constructions of maximal families of MUBs. These are Fourier matrices which we will now introduce.

## 7 Fourier Matrices

Fourier matrix is the most popular type of Hadamard matrices. It is called so because it performs the Fourier transform of a finite Abelian group. Fourier transform is widely used in many areas of mathematics, physics and computer science. However, here we will be mostly interested in one simple property of Fourier matrices. Namely, the rows of the Fourier matrix of a group  $G$  with the Hadamard product operation form a group isomorphic to the group  $G$  (see later).

Let us take an Abelian group  $G = \mathbb{Z}_{d_1} \times \mathbb{Z}_{d_2} \times \cdots \times \mathbb{Z}_{d_m}$  of order  $n = d_1 d_2 \cdots d_m$ . By the structure theorem for finite Abelian groups, each finite Abelian group is isomorphic to a group of this form (see, e.g., [17]).

Later on we will be also interested in the group  $\tilde{G} = \mathbb{R}_{d_1} \times \mathbb{R}_{d_2} \times \cdots \times \mathbb{R}_{d_m}$ , where  $\mathbb{R}_a$  is the group of real numbers modulo  $a$  with the addition operation. Note that  $G_1 \cong G_2$  does not imply  $\tilde{G}_1 \cong \tilde{G}_2$ . Also, the group  $G$  is a subgroup of  $\tilde{G}$ . In addition to that we will use notation  $G^*$  for the set of non-zero elements of  $G$ , and  $\tilde{G}^*$  for the set of elements of  $\tilde{G}$  with at least one component being a non-zero integer.

The Fourier transform usually is defined via the dual group which is formed of all the characters of the group. A character of an Abelian group is its morphism

to the multiplicative group of unit-modulus complex number. It is possible to establish an isomorphism from  $G$  to  $\hat{G}$  (the dual group) by

$$\chi_a(b) = \exp\left(\sum_{j=1}^m \frac{2\pi i}{d_j} a_j b_j\right), \tag{6}$$

where  $a = (a_1, a_2, \dots, a_m)$  and  $b = (b_1, b_2, \dots, b_m)$  are elements of  $G$  and  $\chi_a$  is the element of  $\hat{G}$  corresponding to  $a$ . Note that the expression  $\chi_a(b)$  is symmetric in  $a$  and  $b$ . We will also extend the definition (6) to any  $a$  and  $b$  in  $\tilde{G}$ . The following lemma is a classical result.

**Lemma 11.** *Let  $x$  be an element of  $\tilde{G}$ . Then  $\sum_{y \in G} \chi_y(x) = 0$  if and only if  $x \in \tilde{G}^*$ .*

*Proof.* Let us write  $x = (x_1, x_2, \dots, x_m)$ . We have:

$$\sum_{y \in G} \chi_y(x) = \prod_{j=1}^m \sum_{k=0}^{d_j-1} \exp\left(\frac{2\pi i}{d_j} x_j k\right).$$

Lemma follows from the fact that the roots of the equation  $\sum_{k=0}^{d_j-1} \omega^k = 0$  in  $\omega$  are exactly the roots of unity  $\exp(\frac{2\pi i}{d_j} x_j)$ , with  $x_j$  an integer,  $0 < x_j < d_j$ .  $\square$

**Corollary 12.** *The matrix  $F = (f_{i,j})$ , indexed by the elements of  $G$  and with  $f_{i,j} = \chi_j(i)$ , is an Hadamard matrix.*

*Proof.* Clearly, all elements of the matrix have absolute value 1. The inner product of the rows indexed by  $a$  and  $b$  with  $a \neq b$  is

$$\sum_{y \in G} \overline{\chi_y(a)} \chi_y(b) = \sum_{y \in G} \chi_y(b - a) = 0.$$

Hence, two distinct rows are orthogonal and the matrix  $F$  is Hadamard.  $\square$

Matrix  $F$  from the last corollary is called the *Fourier matrix* of the group  $G$ . As an example, if we take  $G = \mathbb{Z}_n$ , we obtain the matrix

$$\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_n & \omega_n^2 & \dots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \dots & \omega_n^{2n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2n-2} & \dots & \omega_n^{n^2-2n+1} \end{pmatrix}$$

with  $\omega_n = e^{2\pi i/n}$ . An arbitrary Fourier matrix is equal to a tensor product of such matrices.

The Fourier matrix of the group  $G$  has some good properties. At first, it is symmetric. Next, let us denote by  $R_i$  the row that corresponds to the element  $i \in G$ . It is easy to see that  $R_i \circ R_j = R_{i+j}$ . So, the set of rows (the set of columns) forms a group, with the Hadamard multiplication as an operation, that it is isomorphic to the original group  $G$ .

*Remark 13.* Note that the statement of Corollary 12 holds in more general assumptions. Take any subset  $X \subset \tilde{G}$  of size  $|G|$  such that for any  $a, b \in X$  with  $a \neq b$  we have  $a - b \in \tilde{G}^*$ . Then the matrix  $F = (f_{xj})$  ( $x \in X, j \in G$ ), with  $f_{xj} = \chi_j(x)$ , is Hadamard.

For example, if we take  $G = \mathbb{Z}_3 \times \mathbb{Z}_2$  and  $X = \{(0, 0), (0, 1), (1, a), (1, 1 + a), (2, b), (2, 1 + b)\}$  where  $0 \leq a, b \leq 1$  are some reals, then we get a matrix equivalent to one in equation (4) of [4].

## 8 Fourier Matrices in Homogeneous Systems

Let  $H$  be an Hadamard  $n \times n$ -matrix. Denote its rows by  $\{R_i\}, i = 0, 1, \dots, n-1$ . Rescaling of columns does not change the L-graph, so we may always assume that  $R_0$  consists only of ones.

For a fixed  $i$  the set  $\{R_i \circ R_j \mid j = 0, \dots, n-1\}$  is an orthogonal basis of  $\mathbb{C}^n$ . Hence, any  $R_a \circ R_b$  is not orthogonal to at least one of  $\{R_i \circ R_j \mid j = 0, \dots, n-1\}$ . If  $H$  is a Fourier matrix, then  $R_a \circ R_b$  is not orthogonal to exactly one of  $\{R_i \circ R_j \mid j = 0, \dots, n-1\}$ : the one with  $i + j = a + b$ .

And conversely, if any  $R_a \circ R_b$  is not orthogonal to exactly one of  $\{R_0 \circ R_j \mid j = 0, \dots, n-1\}$  then  $L(H)$  is isomorphic to the L-graph of a Fourier matrix. Indeed, let  $G$  be the set of directions (equivalence classes of collinear vectors) defined by rows of  $H$  with the Hadamard product operation. The set is finite, it is closed under the operation,  $R_0$  is the identity element, the operation is commutative and associative and for any fixed  $i$  the operation  $R_j \mapsto R_i \circ R_j$  is a bijection. Hence,  $G$  is a finite Abelian group, and  $L(H)$  is isomorphic to the L-graph of the Fourier matrix of  $G$ . So, we have proved the following result

**Theorem 14.** *Fourier matrices are L-maximal Hadamard matrices. Moreover, their graphs have maximal possible number of edges.*

This result explains why Fourier matrices are so useful in the constructions of MUBs. It can be conjectured that Fourier matrices are the only L-maximal Hadamard matrices.

Let  $G$  be a graph. Recall [10] that the *independence number*  $\alpha(G)$  is the greatest number of vertices that are pairwise disjoint, conversely, the *clique number*  $\omega(G)$  is the greatest number of vertices that are all pairwise connected. The minimal number of colours that can be assigned to the vertices of the graph in such a way that any two adjacent vertices are coloured in different colours, is called the *chromatic number*  $\chi(G)$  of the graph. It is easy to show that  $\chi(G) \geq \omega(G)$ .

It seems worth to mention some constructions that are similar to the notion of L-graphs (i.e., when the adjacency relation on the set of vectors is generated using the orthogonality relation). One known to us example is *Hadamard graph* defined in [20]. The set of vertices of the Hadamard graph  $S(n)$  of order  $n$  is the set of all  $\pm 1$ -component vectors of length  $n$ , and two vectors are adjacent iff they are orthogonal. The famous Hadamard conjecture is equivalent to the statement that  $\omega(S(4n)) = 4n$  for any positive integer  $n$ .

It is clear, that for any Hadamard matrix  $H$  acting on  $\mathbb{C}^n$  the clique number of  $L(H)$  is equal to  $n$ . If  $L(H)$  is a subgraph of the L-graph of a Fourier matrix,

then  $\chi(L(H)) = n$  (the colour of a vertex is the corresponding element of the group). However, it is proved in [14] that there is an exponential gap between  $4n$  and  $\chi(S(4n))$ . So, it is quite possible that for some Hadamard matrix  $H$  we would have  $\chi(L(B)) > n$ , and this matrix cannot be covered by a Fourier matrix.

Another nice property of Fourier matrices (noted to be “striking” in [4]) is that any vector  $v$ , unbiased with respect both to the standard basis and a Fourier matrix, can be collected into a whole unbiased basis. It is easy to explain if one notices that a Fourier matrix  $F$  is symmetric and, hence, also its columns  $\{R_i\}$  form a group with the Hadamard multiplication as the operation. The vector  $v$  can be extended to a basis  $\{R_a \circ v \mid a \in G\}$ , and

$$|\langle R_b | R_a \circ v \rangle| = |\langle R_{b-a} | v \rangle| = \frac{1}{\sqrt{n}}.$$

As mentioned above, all known constructions of complete systems of MUBs are built from Fourier matrices. In the light of Proposition 9 it seems a good choice, since Fourier matrices are L-maximal Hadamard matrices. Moreover, the L-graph of a Fourier matrix covers a fraction of roughly  $\frac{n-1}{n}$  edges of the complete graph, so it remains to find a flat matrix (a more general notion) to cover the remaining fraction of  $\frac{1}{n}$  edges (a less number of edges). It seems that it should be easy, but it is not.

**Proposition 15.** *Let  $H$  be the Fourier matrix of a group  $G$  and  $A$  be a flat matrix with rows  $\{R_i\}_{i \in G}$ . The system defined by (5) is a complete set of MUHs if and only if for any non-zero  $\Delta \in G$  the matrix  $D_\Delta$  with rows from*

$$\{R_{i+\Delta} \circ R_i^{(-1)} \mid i \in G\}$$

*is an Hadamard matrix. (Here  $R_i^{(-1)}$  stands for the element-wise inverse of  $R_i$ . It is the Hadamard  $(-1)$ -st power).*

*Proof.* Suppose we have a complete set of MUHs. It follows from Proposition 9 that

$$\forall g_1, g_2, g_3, g_4 \in G : \left. \begin{array}{l} g_1 + g_2 = g_3 + g_4 \\ \{g_1, g_2\} \neq \{g_3, g_4\} \end{array} \right\} \implies R_{g_1} \circ R_{g_2} \perp R_{g_3} \circ R_{g_4}. \quad (7)$$

Clearly, each element of  $D_\Delta$  is of absolute value 1. Let us take  $g_1 \neq g_3$ . Then

$$\langle R_{g_1+\Delta} \circ R_{g_1}^{(-1)} | R_{g_3+\Delta} \circ R_{g_3}^{(-1)} \rangle = \langle R_{g_1+\Delta} \circ R_{g_3} | R_{g_3+\Delta} \circ R_{g_1} \rangle.$$

Moreover,  $(g_1 + \Delta) + g_3 = (g_3 + \Delta) + g_1$ ,  $g_3 \neq g_1$  and  $g_3 \neq g_3 + \Delta$ . Using (7) with  $g_2 = g_3 + \Delta$  and  $g_4 = g_1 + \Delta$ , we have  $R_{g_1+\Delta} \circ R_{g_1}^{(-1)} \perp R_{g_3+\Delta} \circ R_{g_3}^{(-1)}$ .

The proof of the converse statement is similar. □

Note that Hadamard matrices are quite rare, and here from one flat matrix one should extract  $n - 1$  Hadamards. It explains, to some extent, why it is not so

easy to find a convenient matrix  $A$ . In practice, matrices  $D_\Delta$  are chosen to be (up to some equivalence) equal to the same Fourier matrix. Now we give three possible kinds of restrictions on  $D_\Delta$  and describe the corresponding constructions in terms of functions acting from one Abelian group into another.

Suppose matrix  $H$  (as in (5)) is the Fourier matrix of the group  $G = \mathbb{Z}_{d_1} \times \mathbb{Z}_{d_2} \times \cdots \times \mathbb{Z}_{d_m}$  and let  $N = \mathbb{Z}_{d'_1} \times \mathbb{Z}_{d'_2} \times \cdots \times \mathbb{Z}_{d'_m}$  be the group of the same size. Suppose all matrices  $D_\Delta$  are equal (up to a permutation of rows) to the Fourier matrix  $F$  of  $N$  and each row of  $A$  (that we want to construct) is a row of  $F$ . Define the function  $f : G \rightarrow N$  as assigning to the index of a row of  $A$  the index of the row of  $F$  that stands in this place. It is easy to see that this construction satisfies the condition of Proposition 9 if and only if  $f$  satisfies

$$\forall g_1, g_2, g_3, g_4 \in G : \left. \begin{array}{l} g_1 + g_2 = g_3 + g_4 \\ f(g_1) + f(g_2) = f(g_3) + f(g_4) \end{array} \right\} \implies \{g_1, g_2\} = \{g_3, g_4\}. \quad (8)$$

This is not the most general case. If we allow  $D_\Delta$  to be equal to the matrix  $F$  with row permuted and each column multiplied by  $\chi_x(a)$  where  $a$  is the index of the column and  $x$  is some element of  $\tilde{N}$  (that depends on  $\Delta$ ), then we can take the matrix  $A = (a_{\ell r})$ , ( $\ell \in G, r \in N$ ) defined by  $a_{\ell r} = \chi_r(f(\ell))$ , where function  $f : G \rightarrow \tilde{N}$  satisfies

$$\forall g_1, g_2, g_3, g_4 \in G : \left. \begin{array}{l} g_1 + g_2 = g_3 + g_4 \\ \{g_1, g_2\} \neq \{g_3, g_4\} \end{array} \right\} \implies f(g_1) + f(g_2) - f(g_3) - f(g_4) \in N^*. \quad (9)$$

Finally, from Lemma 11 it follows that this approach gives a complete system of MUHs if and only if  $f : G \rightarrow \tilde{N}$  satisfies

$$\forall g_1, g_2, g_3, g_4 \in G : \left. \begin{array}{l} g_1 + g_2 = g_3 + g_4 \\ \{g_1, g_2\} \neq \{g_3, g_4\} \end{array} \right\} \implies f(g_1) + f(g_2) - f(g_3) - f(g_4) \in \tilde{N}^*. \quad (10)$$

However, in this case matrices  $D_\Delta$  are not longer equivalent to a Fourier matrix, but rather to a matrix mentioned in the remark after Corollary 12.

Summing everything up, we have the following result:

**Theorem 16.** *Condition (10) is more general than the one in (9) that, in its turn, is more general than the one in (8). Formula*

$$(v_k^{(r)})_\ell = \frac{1}{\sqrt{n}} \chi_k(\ell) \chi_r(f(\ell)), \quad (11)$$

*(with  $k, \ell \in G$  and  $r \in N$ ) gives a complete system of MUHs if and only if the function  $f$  (acting from  $G$  to  $\tilde{N}$ ) satisfies (10).*

A similar result appeared in [28]. We postpone a discussion of related topics to Section 10. In the next section we show classical constructions of complete systems of MUBs in the light of Theorem 16.

## 9 Known Constructions

Now we will give two known examples of complete sets of MUBs in the terms of the previous corollary.

Construction essentially corresponding to the following one was first obtained for  $GF(p)$  by Ivanović in [21] and in the general case by Fields and Wootters in [34].

**Lemma 17.** *If  $n = p^k$  is a power of an odd prime, then the function  $f(x) = x^2$  with  $G = N$  being the additive group of  $GF(n)$  (i.e.  $\mathbb{Z}_p^k$ ) satisfies (8).*

*Proof.* Let us suppose  $g_1 + g_2 = g_3 + g_4$  and  $g_1^2 + g_2^2 = g_3^2 + g_4^2$ . Then  $g_1 - g_3 = g_4 - g_2$  and  $(g_1 - g_3)(g_1 + g_3) = (g_4 - g_2)(g_4 + g_2)$ . If  $g_1 = g_3$ , we are done. Otherwise, we can cancel  $g_1 - g_3$  out from the last equality and get  $g_1 + g_3 = g_4 + g_2$ . Together with the first equality it gives  $2(g_2 - g_3) = 0$ . Because 2 does not divide  $n$ ,  $g_2 = g_3$  and we are done.  $\square$

If  $n$  is even we have to be a bit more tricky. Let us remind, that most common construction of finite field  $GF(2^k)$  is of polynomials with degree smaller than  $k$  and coefficients from  $\{0, 1\}$ . All operations are performed modulo 2 and  $h$ , where  $h$  is an irreducible polynomial of degree  $k$ . We will treat these polynomials as integer polynomials. The next lemma also leads to the construction first obtained by Fields and Wootters in [34].

**Lemma 18.** *Let  $G$  be the additive group of  $GF(2^k)$ . Then the function  $f : G \rightarrow \tilde{G}$  defined with*

$$f(x) = \frac{x^2}{2} \pmod{(2, h)}$$

*satisfies (9) with  $N = G$ .*

*Proof.* Suppose  $g_1 + g_2 \equiv g_3 + g_4 \pmod{(2, h)}$ . Then  $(g_1 + g_2)^2 \equiv (g_3 + g_4)^2 \pmod{(2, h)}$ . Hence,  $g_1^2 + g_2^2 - g_3^2 - g_4^2 \equiv 0 \pmod{(2, h)}$ . This means that

$$f(g_1) + f(g_2) - f(g_3) - f(g_4) = \frac{g_1^2 + g_2^2 - g_3^2 - g_4^2}{2} \pmod{(2, h)}$$

is an integer polynomial. The only way it could not belong to  $N^*$  is if it was equal to 0. Let us suppose it is equal to zero and prove that in this case  $\{g_1, g_2\} = \{g_3, g_4\}$ .

Let us define  $s = (g_1 + g_2) \pmod{2}$ . Then also  $g_1^2 + g_2^2 \equiv s^2 \pmod{2}$ . Consider the following equation in  $x$ :

$$\frac{g_1^2 + g_2^2 - x^2 - (s - x)^2}{2} \equiv 0 \pmod{(h, 2)}.$$

Both  $g_1$  and  $g_2$  are its roots. The polynomial of  $x$  can be rewritten as  $\frac{g_1^2 + g_2^2 - s^2}{2} + sx - x^2$ . One may notice that  $\frac{g_1^2 + g_2^2 - s^2}{2}$  is an integer polynomial, so taking it modulo  $h$  and 2 we obtain an equation of the second degree in  $GF(2^k)$ :

$$x^2 - sx - \frac{g_1^2 + g_2^2 - s^2}{2} = 0.$$

If  $g_1 \neq g_2$ , no other element except them can satisfy it. If  $g_1 = g_2$  then  $s = 0$  and this equation has only one root, because  $x \mapsto x^2$  is a bijection in  $GF(2^k)$  (a Frobenius map). Thus,  $f$  satisfy (9).  $\square$

The last two lemmas combine into the following well-known result:

**Theorem 19.** *If  $n$  is a prime power then there exists a complete set of MUBs in  $\mathbb{C}^n$ .*

## 10 Related Combinatorial Structures

Observing formulas (8), (9) and (10) one can conclude that they, especially (8), are of a highly combinatorial nature. It turns out that they indeed have a strong link with some well-studied combinatorial structures.

Suppose  $G$  and  $N$  are Abelian groups with  $|G| \leq |N| < \infty$ . Functions  $f : G \rightarrow N$ , for which the equation  $f(x+a) - f(x) = b$  has no more than 1 solution for all  $a, b \in G$  not equal to zero simultaneously, are called *differentially 1-uniform* [25]. If  $N$  satisfies  $|G|/|N| = m \in \mathbb{N}$  and function  $f : G \rightarrow N$  is such that  $|\{x \in G \mid f(x+a) - f(x) = b\}| = m$  for any  $b \in N$  and non-zero  $a \in G$ , the latter is called *perfect non-linear* [7]. These functions are used in cryptography to construct S-boxes that are not vulnerable to differential cryptanalysis.

If  $|G| = |N|$  as in (8), these two notions coincide and function  $f$  is sometimes called a *planar function*. This name is given because any planar function gives rise to an affine plane [9]. For functions satisfying (9) we will use name *fractional planar*.

The following planar functions from  $GF(p^k)$ , with  $p$  odd, to itself are known:

- $f(x) = x^{p^\alpha + 1}$ , where  $\alpha$  is a non-negative integer with  $k/\gcd(k, \alpha)$  being odd. See [9].
- $f(x) = x^{(3^\alpha + 1)/2}$  only for  $p = 3$ ,  $\alpha$  is odd, and  $\gcd(k, \alpha) = 1$ . See [8].
- $f(x) = x^{10} - ux^6 - u^2x^2$  only for  $p = 3$ ,  $k$  is odd, and  $u$  is a non-zero element of  $GF(p^k)$ . The special case of  $u = -1$  was obtained in [8], the general case is due to [11].

The construction with  $f(x) = x^2$  from the previous section is from the first class.

Let  $K$  again be an Abelian group and  $N$  be its subgroup. A subset  $R \subset K$  is called a *relative  $(m, n, r, \lambda)$ -difference set* if  $|K| = nm$ ,  $|N| = n$ ,  $|R| = r$  and

$$|\{r_1, r_2 \in R \mid r_1 - r_2 = b\}| = \begin{cases} r, & b = 0; \\ 0, & b \in N \setminus \{0\}; \\ \lambda, & b \in K \setminus N. \end{cases}$$

Relative difference set is a generalization of classical difference set and it was introduced in [12]. If  $r = m$  the difference set is called *semiregular*. A relative difference set is called *splitting* if  $K = G \times N$ , i.e. if  $N$  has a complement in  $K$ .

This notion is interesting to us because of the following easy observation (see, e.g., [26]). Let  $G$  and  $N$  be arbitrary finite groups and  $f$  be a function from  $G$  to

$N$ . The set  $\{(x, f(x)) \mid x \in G\}$  is a semiregular splitting ( $|G|, |N|, |G|, |G|/|N|$ )-difference set in  $G \times N$  relative to  $\{1\} \times N$  if and only if  $f$  is perfect nonlinear. Thus, planar functions correspond to splitting relative  $(n, n, n, 1)$ -difference sets. We extend this result a bit:

**Theorem 20.** *Let  $K$  be an Abelian group of size  $n^2$  having a subgroup  $N = \mathbb{Z}_{d_1} \times \mathbb{Z}_{d_2} \times \cdots \times \mathbb{Z}_{d_{m'}}$  of size  $n$ . The following two statements are equivalent:*

- (a) *There exists a semiregular  $(n, n, n, 1)$ -difference set  $R$  in  $K$  relative to  $N$ .*
- (b) *There exists a fractional planar function  $f : G \rightarrow \tilde{N}$  where  $G \cong K/N$ .*

*Proof.* Suppose we have a relative difference set. Fix any  $G = \mathbb{Z}_{d_1} \times \mathbb{Z}_{d_2} \times \cdots \times \mathbb{Z}_{d_m}$  such that  $G \cong K/N$ . Let  $k_1, \dots, k_m \in K$  be representants of the basis of  $G$ . Denote by  $(s_{1i}, s_{2i}, \dots, s_{m'i})$  the element  $d_i k_i \in N$ ,  $i = 1, \dots, m$ .

Define an Abelian group  $K'$  as follows. Its elements are from the direct product  $G \times N$  and the sum of two elements  $(x_1, x_2, \dots, x_m; y_1, y_2, \dots, y_{m'})$  and  $(z_1, z_2, \dots, z_m; t_1, t_2, \dots, t_{m'})$  is defined as  $(a_1, a_2, \dots, a_m; b_1, b_2, \dots, b_{m'})$  where  $a_i = x_i + z_i$  and

$$\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{m'} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{m'} \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_{m'} \end{pmatrix} + \begin{pmatrix} s_{11} & s_{12} & \cdots & s_{1m} \\ s_{21} & s_{22} & \cdots & s_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ s_{m'1} & s_{m'2} & \cdots & s_{m'm} \end{pmatrix} \begin{pmatrix} [x_1 + z_1 \geq d_1] \\ [x_2 + z_2 \geq d_2] \\ \vdots \\ [x_m + z_m \geq d_m] \end{pmatrix} \quad (12)$$

where  $[x_i + z_i \geq d_i]$  is equal to 1 if the sum of  $x_i$  and  $z_i$ , taken as integers, exceeds  $d_i$  and is equal to 0 otherwise. It is not hard to check that  $\varphi : K' \rightarrow K$ , defined with

$$(x_1, x_2, \dots, x_m; y) \mapsto y + \sum_{i=1}^m x_i k_i,$$

is an isomorphism. As usually, we identify elements of  $G$  with the set  $\{(x, 0) \mid x \in G\}$  and  $N$  with  $\{(0; y) \mid y \in N\}$ .

Denote by  $S$  the  $m' \times m$ -matrix whose  $(i, j)$ -th element is equal to  $s_{ij}/d_i$ . Clearly,  $\psi : K' \rightarrow \tilde{N}$  defined by

$$\psi(x, y) = y + Sx$$

is a morphism. Since  $R$  is a semiregular relative difference set, for any  $x \in K/N$  we can find a unique element  $r_x \in R$  with projection on  $K/N$  equal to  $x$ . Define  $f(x) = \psi(\varphi^{-1}(r_x))$ .

Let us prove that  $f$  is fractional planar. At first, note that for any  $x \in G$ :  $\varphi^{-1}(r_x) = (x, y)$  for some  $y \in N$ . Then suppose that  $g_1, g_2, g_3, g_4 \in G$  are such that  $g_3 - g_1 = g_2 - g_4 \neq 0$  and  $g_1 \neq g_4$ . Denote  $(x_1, y_1) = \varphi^{-1}(r_{g_3 - g_1})$  and  $(x_2, y_2) = \varphi^{-1}(r_{g_2 - g_4})$ . We have  $x_1 = x_2$  (because  $g_3 - g_1 = g_2 - g_4$ ) and  $y_1 \neq y_2$  (because  $r_{g_3 - g_1} \neq r_{g_2 - g_4}$ ). From the definition of  $\psi$  we have  $(f(g_3) - f(g_1)) - (f(g_2) - f(g_4)) \in N^*$ .

Suppose conversely that we have a fractional planar function  $f : G \rightarrow \tilde{N}$  with the same expressions for  $G$  and  $N$ . Define function  $\{\cdot\}$  that takes the fractional

part of every component of an element of  $\tilde{N}$ . Define also  $\tilde{f}(x) = \{f(x)\}$ . Then (9) yields

$$(a + b = c + d) \implies (\tilde{f}(a) + \tilde{f}(b) - \tilde{f}(c) - \tilde{f}(d) \in N).$$

Since the condition on  $f$  is invariant under adding a constant to the function, we may assume that  $\tilde{f}(0) = 0$ . Then  $\tilde{f}(a + b) = \{\tilde{f}(a) + \tilde{f}(b)\}$ . Now it is easy to deduce that  $\tilde{f}(x) = \{Sx\}$  where  $S$  is defined in the same way as before for some integers  $s_{ij}$ .

Define  $K'$  as in (12) and define

$$R = \{(x; f(x) - Sx) \mid x \in G\}.$$

Similar reasoning as before shows that  $R$  is semiregular difference set relative to  $N$ .  $\square$

So, we have proved that if matrix  $H$  in (5) is a Fourier matrix, and all  $D_\Delta$  are equivalent (in some sense) Fourier matrices, then the existence of a complete system of MUHs in  $\mathbb{C}^n$  is equivalent to the existence of a relative  $(n, n, n, 1)$ -difference set. In fact, a more general result [16] is known: the existence of a relative  $(n, k, n, \lambda)$ -difference set implies the existence of  $k$  MUHs in  $\mathbb{C}^n$ .

It is proved in [6] that a relative  $(n, n, n, 1)$ -difference set exists only if  $n$  is a prime power. Thus, using the approach with  $f$  satisfying (9) it is not possible to construct a complete system of MUBs for any new dimension. It is still not clear what can be said in the case of general  $D_\Delta$  and, in particular, in the case of  $f$  satisfying (10).

## 11 Conclusion

In this paper we have shown that MUBs stand close to sequences with low correlation, similar constructions and lower bounds can be used in both. In particular, one of the lower bounds (the Welch bounds) gives a nice characterisation of MUBs in terms of orthogonality of a certain collection of vectors. It could be interesting to try to use other constructions and bounds from one area in another.

In particular, it is tempting to use criterion of Theorem 4 to other objects. One example could be SIC-POVMs, because it is proved in [22] that they also attain the Welch bound for  $k = 2$ . Earlier this was proved in [27].

Our criterion seems to have no use for non-complete systems of MUBs. However, in [28] it is proposed to use weighted 2-designs consisting of bases for quantum state estimation when no complete system of MUBs is known. Our criterion is suitable in this case as well. Let us give some details.

The problem is to find such orthonormal bases  $B_0, B_1, \dots, B_k$  of  $\mathbb{C}^n$  and weights  $w_0, w_1, \dots, w_k$  that are non-negative real numbers that the set  $\{w_i x \mid x \in B_i, i = 0, 1, \dots, k\}$  attain the Welch bound for  $k = 2$ . In particular, one of the main results of [28] can be proved similarly to Theorem 16:

**Theorem 21.** *The existence of a differentially 1-uniform function  $f$  from an Abelian group  $G$  into an Abelian group  $N$  with  $|G| = n$  and  $|N| = m$  implies the existence of a weighted 2-design in  $\mathbb{C}^n$  formed from  $m + 1$  orthonormal bases.*

Another direction of future research is the investigation of L-maximal flat and Hadamard matrices in order to find new systems of MUBs or to prove that such systems can be reduced to other already studied cases. A question of non-homogeneous systems of MUBs also remains open.

## Acknowledgements

We would like to thank José Ignacio Rosado for bringing our attention to some errors in the early version of the paper.

## References

1. Aharonov, Y., Englert, B.-G.: The mean kings problem: Spin 1, *Z. Naturforsch.* 56a, 16–19 (2001)
2. Alltop, W.O.: Complex sequences with low periodic correlations. *IEEE Transactions on Information Theory* 26(3), 350–354 (1980)
3. Beth, T., Jungnickel, D., Lenz, H.: *Design Theory*, 2nd edn. Cambridge University Press, Cambridge (1999)
4. Bengtsson, I., Bruzda, W., Ericsson, A., Larsson, J.-A., Tadej, W., Życzkowski, K.: *Mubs and Hadamards of Order Six* (2006) (arXiv:quant-ph/0610161 v1)
5. Bennett, C.H., Brassard, G.: Quantum Cryptography: Public Key Distribution and Coin Tossing. In: *Proceedings of the IEEE International Conference on Computers, Systems and Signal Processing*, New York, pp. 175–179 (1984)
6. Blokhuis, A., Jungnickel, D., Schmidt, B.: Proof of the Prime Power Conjecture for Projective Planes of Order  $n$  with Abelian Collineation Groups of Order  $n^2$ . *Proceedings of AMS* 130(5), 1473–1476 (2001)
7. Carlet, C., Ding, C.: Highly nonlinear mappings. *Journal of Complexity* 20, 205–244 (2004)
8. Coulter, R.S., Matthews, R.W.: Planar functions and planes of Lenz-Barlotti class II. *Des., Codes, Cryptogr.* 10, 167–184 (1997)
9. Dembowski, P., Ostrom, T.G.: Planes of order  $n$  with collineation groups of order  $n^2$ . *Math. Zeitschr.* 103, 239–258 (1968)
10. Diestel, R.: *Graph theory*, 3rd edn. Springer, Heidelberg (2005)
11. Ding, C., Yuan, J.: A family of skew Hadamard difference sets. *Journal of Combinatorial Theory, Series A* 113, 1526–1535 (2006)
12. Elliott, J.E.H., Butson, A.T.: Relative difference sets. *Illinois J. Math.* 10, 517–531 (1966)
13. Englert, B.-G.: Mutually unbiased bases. Problem page in *Quantum Information at TU Braunschweig*, <http://www.imaph.tu-bs.de/qi/problems/13.html>
14. Frankl, P.: Orthogonal vectors in the  $n$ -dimensional cube and codes with missing distances. *Combinatorica* 6(3), 279–285 (1986)
15. Gold, R.: Maximal recursive sequences with 3-valued recursive cross-correlation functions. *IEEE Trans. Inform. Theory* IT-14, 154–156 (1967)

16. Godsil, C., Roy, A.: Equiangular lines, mutually unbiased bases, and spin models (2005) (arXiv:quant-ph/0511004 v2)
17. Hall Jr., M.: The theory of groups. The Macmillan Company, Basingstoke (1968)
18. Helleseeth, T., Kumar, V.J.: Sequences with low correlation. In: Pless, V., Huffman, C. (eds.) Handbook of Coding Theory. Elsevier, Amsterdam (1998)
19. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (1985)
20. Ito, N.: Hadamard graphs. Graphs and Combinatorics 1, 57–64 (1985)
21. Ivanović, I.D.: Geometrical description of quantal state determination. J. Phys. A 14, 3241–3245 (1981)
22. Klappenecker, A., Rötteler, M.: Mutually Unbiased Bases are Complex Projective 2-Designs (2005) (arXiv:quant-ph/0502031 v2)
23. Klappenecker, A., Rötteler, M.: Constructions of Mutually Unbiased Bases (2003) (quant-ph/0309120)
24. Massey, J.L., Mittelholzer, T.: Welch's bound and sequence sets for code-division multiple-access systems. In: Sequences II: Methods in Communication, Security and Computer Sciences, pp. 63–78. Springer, Heidelberg (1993)
25. Nyberg, K.: Differentially uniform mappings for cryptography. In: Helleseeth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 55–64. Springer, Heidelberg (1994)
26. Pott, A.: Nonlinear functions in abelian groups and relative difference sets. Discrete Appl. Math. 138, 177–193 (2004)
27. Renes, J., Blume-Kohout, R., Scott, A.J., Caves, C.: Symmetric Informationally Complete Quantum Measurements. J. Math. Phys. 45, 2171–2180 (2003)
28. Roy, A., Scott, A.J.: Weighted complex projective 2-designs from bases: optimal state determination by orthogonal measurements (2007) (quant-ph/0703025 v2)
29. Stanley, R.: Enumerative Combinatorics, vol. 1. Cambridge University Press, Cambridge (1997)
30. Schwinger, J.: Unitary operator bases. Proc. Nat. Acad. Sci. U.S.A. 46, 570–579 (1960)
31. Tadey, W., Zyczkowski, K.: A concise guide to complex Hadamard matrices (2006) (quant-ph/0512154 v2)
32. Waldron, S.: Generalized Welch Bound Equality Sequences Are Tight Frames. IEEE Transactions on Information Theory 49(9), 2307–2309 (2003)
33. Welch, L.R.: Lower bounds on the maximum cross correlations of signals. IEEE Transactions on Information Theory 20(3), 397–399 (1974)
34. Wootters, W.K., Fields, B.D.: Optimal state-determination by mutually unbiased measurements. Annals of Physics 191, 363–381 (1989)
35. Wootters, W.K.: Picturing qubits in phase space. IBM Journal of Research and Development 48(1), 99–110 (2004)
36. Zauner, G.: Quantendesigns – Grundzüge einer nichtkommutativen Designtheorie (in German). PhD thesis, Universität Wien (1999)

# An Efficient Quantum Algorithm for the Hidden Subgroup Problem over Weyl-Heisenberg Groups

Hari Krovi and Martin Rötteler

NEC Laboratories America  
4 Independence Way, Suite 200  
Princeton, NJ 08540, U.S.A.  
{krovi,mroetteler}@nec-labs.com

**Abstract.** Many exponential speedups that have been achieved in quantum computing are obtained via hidden subgroup problems (HSPs). We show that the HSP over Weyl-Heisenberg groups can be solved efficiently on a quantum computer. These groups are well-known in physics and play an important role in the theory of quantum error-correcting codes. Our algorithm is based on non-commutative Fourier analysis of coset states which are quantum states that arise from a given black-box function. We use Clebsch-Gordan decompositions to combine and reduce tensor products of irreducible representations. Furthermore, we use a new technique of changing labels of irreducible representations to obtain low-dimensional irreducible representations in the decomposition process. A feature of the presented algorithm is that in each iteration of the algorithm the quantum computer operates on two coset states simultaneously. This is an improvement over the previously best known quantum algorithm for these groups which required four coset states.

**Keywords:** quantum algorithms, hidden subgroup problem, coset states.

## 1 Introduction

Exponential speedups in quantum computing have hitherto been shown for only a few classes of problems, most notably for problems that ask to extract hidden features of certain algebraic structures. Examples for this are hidden shift problems [DHI03], hidden non-linear structures [CSV07], and hidden subgroup problems (HSPs). The latter class of hidden subgroup problems has been studied quite extensively over the past decade. There are some successes such as the efficient solution of the HSP for any abelian group [Sho97, Kit97, BH97, ME98], including factoring and discrete log as well as Pell's equation [Hal02], and efficient solutions for some non-abelian groups [FIM+03, BCD05]. Furthermore, there are some partial successes for some non-abelian groups such as the dihedral groups [Reg04, Kup05] and the affine groups [MRRS04]. Finally, it has been established that for some groups, including the symmetric group which is connected to the graph isomorphism problem, a straightforward approach requires a

rather expensive quantum processing in the sense that entangling operations on a large number of quantum systems would be required [HMR+06]. What makes matters worse, there are currently no techniques, or even promising candidates for techniques, to implement these highly entangling operations.

The present paper deals with the hidden subgroup problem for a class of non-abelian groups that—in a precise mathematical sense that will be explained below—is not too far away from the abelian case, but at the same time has some distinct non-abelian features that make the HSP over these groups challenging and interesting.

The hidden subgroup problem is defined as follows: we are given a function  $f : G \rightarrow S$  from a group  $G$  to a set  $S$ , with the additional promise that  $f$  takes constant and distinct values on the left cosets  $gH$ , where  $g \in G$ , of a subgroup  $H \leq G$ . The task is to find a generating system of  $H$ . The function  $f$  is given as a black-box, i. e., it can only be accessed through queries and in particular whose structure cannot be further studied. The input size to the problem is  $\log |G|$  and for a quantum algorithm solving the HSP to be efficient means to have a running time that is  $\text{poly}(\log |G|)$  in the number of quantum operations as well as in the number of classical operations.

We will focus on a particular approach to the HSP which proved to be successful in the past, namely the so-called *standard method*, see [GSVV04]. Here the function  $f$  is used in a special way, namely it is used to generate *coset states* which are states of the form  $1/\sqrt{|H|} \sum_{h \in H} |gh\rangle$  for random  $g \in G$ . The task then becomes to extract a generating system of  $H$  from a polynomial number of coset states (for random values of  $g$ ).

A basic question about coset states is how much information about  $H$  they indeed convey and how this information can be extracted from suitable measurements.<sup>1</sup> A fixed POVM  $\mathcal{M}$  operates on a fixed number  $k$  of coset states at once and if  $k \geq 2$  and  $\mathcal{M}$  does not decompose into measurements of single copies, we say that the POVM is an entangled measurement. As in [HMR+06], we call the parameter  $k$  the “jointness” of the measurement. It is known that information-theoretically for any group  $G$  jointness  $k = O(\log |G|)$  is sufficient [EHK04]. While the true magnitude of the required  $k$  can be significantly smaller (abelian groups serve as examples for which  $k = 1$ ), there are cases for which indeed a high order of  $k = \Theta(\log |G|)$  is sufficient *and* necessary. Examples for such groups are the symmetric groups [HMR+06]. However, on the more positive side, it is known that some groups require only a small, sometimes even only constant, amount of jointness. Examples are the Heisenberg groups of order  $p^3$  for a prime  $p$  for which  $k = 2$  is sufficient [BCD05, Bac08a]. In earlier work [ISS07], it has been shown that for the Weyl-Heisenberg groups order  $p^{2n+1}$ ,  $k = 4$  is sufficient [ISS07].

The goal of this paper is to show that in the latter case the jointness can be improved. We give a quantum algorithm which is efficient in the input size (given by  $\log p$  and  $n$ ) and which only requires a jointness of  $k = 2$ .

---

<sup>1</sup> Recall that the most general way to extract classical information from quantum states is given by means of positive operator valued measures (POVMs) [NC00].

**Our results and related work:** The family of groups which we consider in the present paper are well-known in quantum information processing under the name of generalised Pauli groups or Weyl-Heisenberg groups [NC00]. Their importance in quantum computing stems from the fact that they are used to define stabilizer codes [CRSS97, Got96, CRSS98], the class of codes most widely used for the construction of quantum error-correcting codes.

In a more group-theoretical context, the Weyl-Heisenberg groups are known as extraspecial  $p$ -groups (actually, they constitute one of the two families of extraspecial  $p$ -groups [Hup83]). A polynomial-time algorithm for the HSP for the extraspecial  $p$ -groups was already given by Ivanyos, Sanselme, and Santha, [ISS07]. Our approach differs to this approach in two aspects: first, our approach is based on Fourier sampling for the non-abelian group  $G$ . Second, and more importantly, we show that the jointness  $k$ , i. e., the number of coset states that the algorithm has to operate jointly on, can be reduced from  $k = 4$  to  $k = 2$ . Crucial for our approach is the fact that in the Weyl-Heisenberg group the labels of irreducible representations can be changed. This in turn can be used to “drive” Clebsch-Gordan decompositions in such a way that low-dimensional irreducible representations occur in the decomposition.

It is perhaps interesting to note that for the Weyl-Heisenberg groups the states that arise after the measurement in the Fourier sampling approach (also called Fourier coefficients) are typically of a very large rank (i. e., exponential in the input size). Generally, large rank usually is a good indicator of the intractability of the HSP, such as in case of the symmetric group when  $H$  is a full support involution. Perhaps surprisingly, in the case of the Weyl-Heisenberg group it still is possible to extract  $H$  efficiently even though the Fourier coefficients have large rank. We achieve this at the price of operating on two coset states at the same time. This leaves open the question whether  $k = 1$  is possible, i. e., if the hidden subgroup  $H$  can be identified from measurements on single coset states. We cannot resolve this question but believe that this will be hard. Our reasoning is as follows. Having Fourier coefficients of large rank implies that the random basis method [RRS05, Sen06] cannot be applied. The random basis method is a method to derive algorithms with  $k = 1$  whose quantum part can be shown to be polynomial, provided that the rank of the Fourier coefficients is constant.<sup>2</sup> Based on this we therefore conjecture that any efficient quantum algorithm for the extraspecial groups will require jointness of  $k \geq 2$ .

Finally, we mention that a similar method to combine the two registers in each run of the algorithm has been used by Bacon [Bac08a] to solve the HSP in the Heisenberg groups of order  $p^3$ . The method uses a Clebsch-Gordan transform which is a unitary transform that decomposes the tensor product of two irreducible representations [Ser77] into its constituents. The main difference between the Heisenberg group and the Weyl-Heisenberg groups is that the Fourier coefficients are no longer pure states and are of possibly high rank.

---

<sup>2</sup> This can be obtained by combining the random basis method [Sen06] with the de-randomization results of [AE07].

**Organization of the paper:** In Section 2 we review the Weyl-Heisenberg group and its subgroup structure. The Fourier sampling approach and the so-called standard algorithm are reviewed in Section 3. In Section 4 we provide necessary facts about the representation theory that will be required in the subsequent parts. The main result of this paper is the quantum algorithm for the efficient solution of the HSP in the Weyl-Heisenberg groups presented in Section 5. Finally, we offer conclusions in Section 6.

## 2 The Weyl-Heisenberg Groups

We begin by recalling some basic group-theoretic notions. Recall that the center  $Z(G)$  of a group  $G$  is defined as the set of elements which commute with every element of the group i.e.,  $Z(G) = \{c : [c, g] = cgc^{-1}g^{-1} = e \text{ for all } g \in G\}$ , where  $e$  is the identity element of  $G$ . The derived (or commutator) subgroup  $G'$  is generated by elements of the type  $[a, b] = aba^{-1}b^{-1}$ , where  $a, b \in G$ . The reader is invited to recall the definition of semidirect products  $G = N \rtimes H$ , see for instance [Hup83, Ser77]. In the following we give a definition of the Weyl-Heisenberg groups as a semidirect product and give two alternative ways of working with these groups.

**Definition 1.** *Let  $p$  be a prime and let  $n$  be an integer. The Weyl-Heisenberg group of order  $p^{2n+1}$  is defined as the semidirect product  $\mathbb{Z}_p^{n+1} \rtimes_{\phi} \mathbb{Z}_p^n$ , where the action  $\phi$  in the semidirect product is defined on  $x = (x_1, \dots, x_n) \in \mathbb{Z}_p^n$  as the  $(n + 1) \times (n + 1)$  matrix given by*

$$\phi(x) = \begin{pmatrix} 1 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 \\ & \ddots & \ddots & \\ 0 & \dots & 1 & 0 \\ x_1 & x_2 & \dots & x_n & 1 \end{pmatrix}. \tag{1}$$

Any group element of  $\mathbb{Z}_p^{n+1} \rtimes_{\phi} \mathbb{Z}_p^n$  can be written as a triple  $(x, y, z)$  where  $x$  and  $y$  are vectors of length  $n$  whose entries are elements of  $\mathbb{Z}_p$  and  $z$  is in  $\mathbb{Z}_p$ . To relate this triple to the semidirect product, one can think of  $(y, z) \in \mathbb{Z}_p^{n+1}$  and  $x \in \mathbb{Z}_p^n$ . Then, the product of two elements in this group can be written as

$$(x, y, z) \cdot (x', y', z') = (x + x', y + y', z + z' + x' \cdot y), \tag{2}$$

where  $x \cdot y = \sum_i x_i y_i$  is the dot product of two vectors (denoted as  $xy$  in the rest of the paper).

**Fact 1.** [Hup83] *For any  $p$  prime, and  $n \geq 1$ , the Weyl-Heisenberg group is an extraspecial  $p$  group. Recall that a group  $G$  is extraspecial if  $Z(G) = G'$ , the center is isomorphic to  $\mathbb{Z}_p$ , and  $G/G'$  is a vector space.*

Up to isomorphism, extraspecial  $p$ -groups are of two types: groups of exponent  $p$  and groups of exponent  $p^2$ . The Weyl-Heisenberg groups are the extraspecial

$p$ -groups of exponent  $p$ . It was shown in [ISS07] that an algorithm to find hidden subgroups in the groups of exponent  $p$  can be used to find hidden subgroups in groups of exponent  $p^2$ . Therefore, it is enough to solve the HSP in groups of exponent  $p$ . In this paper, we present an efficient algorithm for the HSP over groups of exponent  $p$ .

*Realization via matrices over  $\mathbb{Z}_p$ :* First, we recall that the Heisenberg group of order  $p^3$  (which is the group of  $3 \times 3$  upper triangular matrices with ones on the main diagonal and other entries in  $\mathbb{Z}_p$ ) is a Weyl-Heisenberg group and can be regarded as the semidirect product  $\mathbb{Z}_p^2 \rtimes \mathbb{Z}_p$ . An efficient algorithm for the HSP over this group is given in [BCD05]. Elements of this group are of the type

$$\begin{pmatrix} 1 & y & z \\ 0 & 1 & x \\ 0 & 0 & 1 \end{pmatrix}. \tag{3}$$

The product of two such elements is

$$\begin{pmatrix} 1 & y & z \\ 0 & 1 & x \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & y' & z' \\ 0 & 1 & x' \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & y + y' & z + z' + x'y \\ 0 & 1 & x + x' \\ 0 & 0 & 1 \end{pmatrix} \tag{4}$$

Thus, such a matrix can be identified with a triple  $(x, y, z)$  in  $\mathbb{Z}_p^2 \rtimes \mathbb{Z}_p$ . This matrix representation of the Heisenberg group can be generalized for any  $n$ . We can associate a triple  $(x, y, z)$  where  $x, y \in \mathbb{Z}_p^n$  and  $z \in \mathbb{Z}_p$  with the  $(n+2) \times (n+2)$  matrix

$$\begin{pmatrix} 1 & y_1 & \dots & y_n & z \\ 0 & 1 & \dots & 0 & x_1 \\ \vdots & \vdots & \dots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & x_n \\ 0 & 0 & \dots & 0 & 1 \end{pmatrix}. \tag{5}$$

*Realization via unitary representation:* Finally, there is another useful way to represent the Weyl-Heisenberg group. The  $n$  qubit Pauli matrices form a faithful (irreducible) representation of the Weyl-Heisenberg  $p$ -group. For any  $k \neq 0$ , we can associate with any triple  $(x, y, z)$  in  $\mathbb{Z}_p^{n+1} \rtimes \mathbb{Z}_p^n$ , the following matrix:

$$\rho_k(x, y, z) = \omega_p^{kz} X^x Z_k^y, \tag{6}$$

where the matrix  $X = \sum_{u \in \mathbb{Z}_p^n} |u + 1\rangle \langle u|$  is the generalized  $X$  operator and the matrix  $Z_k = \sum_{u \in \mathbb{Z}_p^n} \omega_p^k |u\rangle \langle u|$  is the generalized  $Z$  operator, see e. g. [NC00].

*Subgroup structure:* In the following we will write  $G$  in short for Weyl-Heisenberg groups. Using the notation introduced above the center  $Z(G)$  (or  $G'$ ) is the group  $Z(G) = \{(0, 0, z) | z \in \mathbb{Z}_p\}$  and is isomorphic to  $\mathbb{Z}_p$ . As mentioned above, the quotient group  $G/G'$  is a vector space isomorphic to  $\mathbb{Z}_p^{2n}$ . This space can be

regarded as a *symplectic* space with the following inner product:  $(x, y) \cdot (x', y') = (x \cdot y' - y \cdot x')$ , where  $x, y, x', y' \in \mathbb{Z}_p^n$ . The quotient map is just the restriction of the triple  $(x, y, z) \in G$  to the pair  $(x, y) \in \mathbb{Z}_p^{2n}$ . From Eq. (2), it follows that two elements commute if and only if  $xy' - yx' = 0$ . Denote the set of  $(x, y)$  pairs occurring in  $H$  as  $S_H$  i.e., for each triple  $(x, y, z) \in H$ , we have that  $(x, y) \in S_H$  and so  $|S_H| \leq |H|$ . It can be easily verified that  $S_H$  is a vector space and is in fact, a subspace of  $\mathbb{Z}_p^{2n}$ . Indeed, for two elements  $(x, y), (x', y') \in S_H$ , pick two elements  $(x, y, z), (x', y', z') \in H$  and so  $(x+x', y+y', z+z'+x'y) \in H$ . Therefore,  $(x+x', y+y') \in S_H$ . To show that if  $(x, y) \in S_H$ , then  $(ax, ay) \in S_H$  for any  $a \in \mathbb{Z}_p$ , observe that if  $(x, y, z) \in H$ , then  $(x, y, z)^a = (ax, ay, az + \frac{a(a-1)}{2}xy) \in H$ . Therefore,  $(ax, ay) \in S_H$  (in fact, it can be shown that  $S_H \simeq HG'/G'$ , but we do not need this result.) Therefore,  $H \leq G$  is abelian if and only if  $\forall (x, y), (x', y') \in S_H$ , we have that  $xy' - x'y = 0$ . Such a space where all the elements are orthogonal to each other is called *isotropic*.

Now, we make a few remarks about the conjugacy class of some subgroup  $H$ . Consider conjugating  $H$  by some element of  $G$ , say  $g = (x', y', z')$ . For any  $h = (x, y, z) \in H$ , we obtain

$$\begin{aligned} g^{-1}hg &= (-x', -y', -z' + x'y')(x, y, z)(x', y', z') \\ &= (-x', -y', -z' + x'y')(x + x', y + y', z + z' + x'y) \\ &= (x, y, z + x'y - xy') \in H^g. \end{aligned} \tag{7}$$

From this we see that  $S_{H^g} = S_H$ . We show next that  $S_H$  actually characterizes the conjugacy class of  $H$ . Before proving this result we need to determine the stabilizer of  $H$ . The stabilizer  $H_S$  of  $H$  is defined as the set of elements of  $G$  which preserve  $H$  under conjugation i.e.,  $H_S = \{g \in G | H^g = H\}$ . From Eq. (7), we can see that  $g = (x', y', z') \in H_S$  if and only if  $x'y - xy' = 0$  for all  $(x, y, z) \in H$ . Thus, the stabilizer is a group such that  $S_{H_S} = S_H^\perp$ , where  $S_H^\perp$  is the orthogonal space under the symplectic inner product defined above, i.e.,  $H_S = \{(x, y, z) \in G | (x, y) \in S_H^\perp, z \in \mathbb{Z}_p\}$ . In other words, it is obtained by appending the pairs  $(x, y) \in S_H^\perp$  with every possible  $z \in \mathbb{Z}_p$ . Therefore,  $|H_S| = |G'| \cdot |S_H^\perp|$ . Now, we can prove the following lemma.

**Lemma 1.** *Two subgroups  $H_1$  and  $H_2$  are conjugate if and only if  $S_{H_1} = S_{H_2}$ .*

*Proof.* We have already seen that if  $H_1$  and  $H_2$  are conjugates, then  $S_{H_1} = S_{H_2}$ . To show the other direction, we use a counting argument i.e., we show that the number of subgroups  $H'$  of  $G$  such that  $S_{H'} = S_H$  is equal to the number of conjugates of  $H$ . First, assume that the dimension of the vector space  $S_{H_1}$  is  $k$ . Now, the number of conjugates of  $H_1$  is the index of the stabilizer of  $H_1$ . From the above result, the stabilizer has a size  $|G'| |S_{H_1}^\perp| = p \cdot p^{2n-k}$ . Therefore, the index or the number of conjugates of  $H_1$  are  $p^{2n+1}/p^{2n-k-1} = p^k$ . Now, the number of different possible subgroups  $H$  such that  $S_H = S_{H_1}$  is  $p^k$  since each of the  $k$  basis vectors of  $S_{H_1}$  are generators of the subgroup and they can have any  $z$  component independent of each other i.e., there are  $p$  possible choices of  $z$  for each of the  $k$  generators.

The property  $G' = Z(G)$  will be useful in that it will allow us to consider only a certain class of hidden subgroups. We show next that it is enough to consider hidden subgroups which are abelian and do not contain  $G'$ . Recall that  $H$  is normal in  $G$  (denoted  $H \trianglelefteq G$ ) if  $g^{-1}hg \in H$  for all  $g \in G$  and  $h \in H$ .

**Lemma 2.** *If  $G' \leq H$ , then  $H \trianglelefteq G$ .*

*Proof.* Since  $G'$  is the commutator subgroup, for any  $g_1, g_2 \in G$ , there exists  $g' \in G'$  such that  $g_1g_2 = g_2g_1g'$ . Now, let  $h \in H$  and  $g \in G$ . We have  $g^{-1}hg = hg'$  for some  $g' \in G'$ . But since  $G' \leq H$ ,  $hg' = h'$ , for some  $h' \in H$ . Therefore,  $g^{-1}hg = h'$  and hence  $H \trianglelefteq G$ .

**Lemma 3.** *If  $H$  is non-abelian, then  $H \trianglelefteq G$ .*

*Proof.* Let  $h_1, h_2 \in H$  such that  $h_1h_2 \neq h_2h_1$ . Then  $h_1h_2 = h_2h_1g'$  for some  $g' \in G'$  such that  $g' \neq e$ , where  $e$  is the identity element of  $G$ . This means that  $g' \in H$ . Since  $G'$  is cyclic of prime order, it can be generated by any  $g' \neq e$  and hence, we have  $G' \leq H$ . Now, Lemma 2 implies that  $H \trianglelefteq G$ .

From these two lemmas, we have only two cases to consider for the hidden subgroup  $H$ : (a)  $H$  is abelian and does not contain  $G'$  and (b)  $H$  is normal in  $G$ . It is possible to tell the cases apart by querying the hiding function  $f$  twice and checking whether  $f(e)$  and  $f(g')$  are equal for some  $g' \neq e$  and  $g' \in G'$ . If they are equal then  $G' \leq H$  and  $H \trianglelefteq G$ , otherwise  $H$  is abelian. If  $H$  is normal, then one can use the algorithm of [HRT03], which is efficient if one can intersect kernels of the irreducible representations (irreps) efficiently. For the Weyl-Heisenberg group, the higher dimensional irreps form a faithful representation and hence do not have a kernel. Thus, when the hidden subgroup is normal, only one dimensional irreps occur and their kernels can be intersected efficiently and the hidden subgroup can be found using the algorithm of [HRT03]. Therefore, we can consider only those hidden subgroups which are abelian and moreover do not contain  $G'$ .

Now, we restrict our attention to the case of abelian  $H$ . Finally, we need the following two results.

**Lemma 4.** *If  $H$  is an abelian subgroup which does not contain  $G'$ , then  $|S_H| = |H|$ .*

*Proof.* Suppose that for some  $(x, y) \in S_H$  there exist two different elements  $(x, y, z_1)$  and  $(x, y, z_2)$  in  $H$ , then by multiplying one with the inverse of the other we get  $(0, 0, z_1 - z_2)$ . Since  $z_1 - z_2 \neq 0$ , this generates  $G'$ , but by our assumption on  $H$ ,  $G' \not\leq H$ . Therefore,  $|S_H| = |H|$ .

The following theorem applies to the case when  $p > 2$ .

**Lemma 5.** *Let  $H$  be an abelian subgroup which does not contain  $G'$ . There exists a subgroup  $H_0$  conjugate to  $H$ , where  $H_0 = \{(x, y, xy/2) | (x, y) \in S_H\}$ .*

*Proof.* We can verify that  $H_0$  is a subgroup by considering elements  $(x, y, xy/2)$  and  $(x', y', x'y'/2)$  in  $H_0$ . Their product is

$$\begin{aligned} (x, y, xy/2) \cdot (x', y', x'y'/2) &= (x + x', y + y', xy/2 + x'y'/2 + x'y) \\ &= (x + x', y + y', xy/2 + x'y'/2 + (x'y + xy')/2) \\ &= (x + x', y + y', (x + x')(y + y')/2), \end{aligned} \quad (8)$$

which is an element of  $H_0$ . Here, we have used the fact that  $H$  is abelian i.e.,  $xy' - x'y = 0, \forall(x, y), (x', y') \in S_H$ . Now for  $H_0$ , since  $S_{H_0} = S_H$ ,  $H_0$  is conjugate to  $H$  using Lemma 1.

Note that  $H_0$  can be thought of as a representative of the conjugacy class of  $H$  since it can be uniquely determined from  $S_H$ . The above lemma does not apply for the case  $p = 2$ . When  $p = 2$ , we have that  $(x, y, z)^2 = (2x, 2y, 2z + xy) = (0, 0, xy)$ . But since we assume that  $G' \not\leq H$ , when  $p = 2$  we must have that  $xy = 0, \forall(x, y, z) \in H$ .

### 3 Fourier Sampling Approach to HSP

We recall some basic facts about the Fourier sampling approach to the HSP, see also [GSVV04, HMR+06]. First, we recall some basic notions of representation theory of finite groups [Ser77] that are required for this approach. Let  $G$  be a finite group, let  $\mathbb{C}[G]$  to denote its group algebra, and let  $\hat{G}$  be the set of irreducible representations (irreps) of  $G$ . We will consider two distinguished orthonormal vector space bases for  $\mathbb{C}[G]$ , namely, the basis given by the group elements on the one hand (denoted by  $|g\rangle$ , where  $g \in G$ ) and the basis given by normalized matrix coefficients of the irreducible representations of  $G$  on the other hand (denoted by  $|\rho, i, j\rangle$ , where  $\rho \in \hat{G}$ , and  $i, j = 1, \dots, d_\rho$  for  $d_\rho$ , where  $d_\rho$  denotes the dimension of  $\rho$ ). Now, the quantum Fourier transform over  $G$ ,  $\text{QFT}_G$  is the following linear transformation [Bet87, GSVV04]:

$$|g\rangle \mapsto \sum_{\rho \in \hat{G}} \sqrt{\frac{d_\rho}{|G|}} \sum_{i, j=1}^{d_\rho} \rho_{ij}(g) |\rho, i, j\rangle. \quad (9)$$

An easy consequence of Schur's Lemma is that  $\text{QFT}_G$  is a unitary transformation in  $\mathbb{C}^{|G|}$ , mapping from the basis of  $|g\rangle$  to the basis of  $|\rho, i, j\rangle$ . For a subgroup  $H \leq G$  and irrep  $\rho \in \hat{G}$ , define  $\rho(H) := \frac{1}{|H|} \sum_{h \in H} \rho(h)$ . Again from Schur's Lemma we obtain that  $\rho(H)$  is an orthogonal projection to the space of vectors that are point-wise fixed by every  $\rho(h), h \in H$ .

Define  $r_\rho(H) := \text{rank}(\rho(H))$ ; then  $r_\rho(H) = 1/|H| \sum_{h \in H} \chi_\rho(h)$ , where  $\chi_\rho$  denotes the character of  $\rho$ . For any subset  $S \leq G$  define  $|S\rangle := 1/\sqrt{|S|} \sum_{s \in S} |s\rangle$  to be the uniform superposition over the elements of  $S$ .

The *standard method* [GSVV04] starts from  $1/\sqrt{|G|} \sum_{g \in G} |g\rangle|0\rangle$ . It then queries  $f$  to get the superposition  $1/\sqrt{|G|} \sum_{g \in G} |g\rangle|f(g)\rangle$ . The state becomes

a mixed state given by the density matrix  $\sigma_H^G = \frac{1}{|G|} \sum_{g \in G} |gH\rangle\langle gH|$  if the second register is ignored. Applying  $\text{QFT}_G$  to  $\sigma_H^G$  gives the density matrix

$$\frac{|H|}{|G|} \bigoplus_{\rho \in \hat{G}} \bigoplus_{i=1}^{d_\rho} |\rho, i\rangle\langle \rho, i| \otimes \rho^*(H),$$

where  $\rho^*(H)$  operates on the space of column indices of  $\rho$ . The probability distribution induced by this base change is given by  $P(\text{observe } \rho) = \frac{d_\rho |H| r_\rho(H)}{|G|}$ . It is easy to see that measuring the rows does not furnish any new information: indeed, the distribution on the row indices is a uniform distribution  $1/d_\rho$ . The reduced state on the space of column indices on the other hand can contain information about  $H$ : after having observed an irrep  $\rho$  and a row index  $i$ , the state is now collapsed to  $\rho^*(H)/r_\rho(H)$ . From this state we can try to obtain further information about  $H$  via subsequent measurements.

Finally, we mention that Fourier sampling on  $k \geq 2$  registers can be defined in a similar way. Here one starts off with  $k$  independent copies of the coset state and applies  $\text{QFT}_G^{\otimes k}$  to it. In the next section, we describe the representation theory of the Weyl-Heisenberg groups. An efficient implementation of  $\text{QFT}_G$  is shown in Appendix A.

## 4 The Irreducible Representations

In this section, we discuss the representation theory of  $G$ , where  $G \cong \mathbb{Z}_p^{n+1} \times \mathbb{Z}_p$  is a Weyl-Heisenberg group. From the properties of being an extraspecial group, it is easy to see that  $G$  has  $p^{2n}$  one dimensional irreps and  $p-1$  irreps of dimension  $p^n$ . The one dimensional irreps are given by

$$\chi_{a,b}(x, y, z) = \omega_p^{(ax+by)}, \quad (10)$$

where  $\omega_p = e^{2\pi i/p}$  and  $a, b \in \mathbb{Z}_p^n$ . Note that

$$\chi_{a,b}(H) = \frac{1}{|H|} \sum_{(x,y,z) \in H} \omega_p^{ax+by} = \frac{1}{|S_H|} \sum_{(x,y) \in S_H} \omega_p^{ax+by}. \quad (11)$$

Since  $S_H$  is a linear space, this expression is non-zero if and only if  $a, b \in S_H^\perp$ . Suppose we perform a QFT on a coset state and measure an irrep label. Furthermore, suppose that we obtain a one dimensional irrep (although the probability of this is exponentially small as we show in the next section). Then this would enable us to sample from  $S_H^\perp$ . If this event of sampling one dimensional irreps would occur some  $O(n)$  times, we would be able to compute a generating set of  $S_H^\perp$  with constant probability. This gives us information about the conjugacy class of  $H$  and from knowing this, it is easy to see that generators for  $H$  itself can be inferred by means of solving a suitable abelian HSP.

Thus, obtaining one dimensional irreps would be useful. Of course we cannot assume to sample from one dimensional irreps as they have low probability of

occurring. Our strategy will be to “manufacture” one dimensional irreps from combining higher-dimensional irreps. First, recall that the  $p^n$  dimensional irreps are given by

$$\rho_k(x, y, z) = \sum_{u \in \mathbb{Z}_p^n} \omega_p^{k(z+yu)} |u+x\rangle \langle u|, \quad (12)$$

where  $k \in \mathbb{Z}_p$  and  $k \neq 0$ . This representation is a faithful irrep and its character is given by  $\chi_k(g) = 0$  for  $g \neq e$  and  $\chi_k(e) = p^n$ . In particular,  $\chi_k(H) = p^n/|H|$ .

The probability of a high dimensional irrep occurring in Fourier sampling is very high (we compute this in Section 5). We consider the tensor product of two such high dimensional irreps. This tensor product can be decomposed into a direct sum of irreps of the group. A unitary base change which decomposes such a tensor product into a direct sum of irreps is called a *Clebsch-Gordan* transform, denoted by  $U_{CG}$ . Clebsch-Gordan transforms have been used implicitly to bound higher moments of a random variable that describes the probability distribution of a POVM on measuring a Fourier coefficient. They have also been used in [Bac08a] to obtain a quantum algorithm for the HSP over Heisenberg groups of order  $p^3$ , and in [Bac08b] for the HSP in the groups  $D_4^n$  as well as for Simon’s problem. Our use of Clebsch-Gordan transforms will be somewhat similar.

For the Weyl-Heisenberg group  $G$ , the irreps that occur in the Clebsch-Gordan decomposition of the tensor product of high dimensional irreps  $\rho_k(g) \otimes \rho_l(g)$  depend on  $k$  and  $l$ . The Clebsch-Gordan transform is given by

$$U_{CG} : |u, v\rangle \rightarrow \sum_{w \in \mathbb{Z}_p^n} \omega_p^{\frac{1}{2}(u+v)w} |u-v, w\rangle. \quad (13)$$

If  $k+l \neq 0$ , then only one irrep of  $G$  occurs with multiplicity  $p^n$ , namely

$$\rho_k(g) \otimes \rho_l(g) \xrightarrow{U_{CG}} I_{p^n} \otimes \rho_{k+l}(g). \quad (14)$$

If  $k+l = 0$ , then all the one dimensional irreps occur with multiplicity one i.e.,

$$\rho_k(g) \otimes \rho_l(g) \xrightarrow{U_{CG}} \bigoplus_{a,b \in \mathbb{Z}_p} \chi_{a,b}(g). \quad (15)$$

Note, however, that the state obtained after Fourier sampling is not  $\rho_k(g) \otimes \rho_l(g)$ , but rather  $\rho_k(H) \otimes \rho_l(H)$ . When we apply the Clebsch-Gordan transform to this state, we obtain one dimensional irreps  $\chi_{a,b}(H)$  on the diagonal. Applying this to  $\rho_{-l}(H) \otimes \rho_l(H)$  gives us

$$\begin{aligned} & \sum_{\substack{(x,y,z), (x',y',z') \in H \\ u,v,w_1,w_2 \in \mathbb{Z}_p^n}} \omega_p^{-l(yu+z)+l(y'v+z')+\frac{1}{2}((u+v)(w_1-w_2)+w_1(x+x'))} \\ & \quad |u-v+x-x', w_1\rangle \langle u-v, w_2| \\ = & \sum_{\substack{(x,y,z), (x',y',z') \in H \\ u',w_1,w_2 \in \mathbb{Z}_p^n}} \omega_p^{\frac{1}{2}(-(y+y')u'+2(z'-z)+w_1(x+x'))} \\ & \quad \sum_{v'} \omega_p^{\frac{1}{2}(v'(w_1-w_2+y'-y))} |u'+x-x', w_1\rangle \langle u', w_2|, \end{aligned}$$

where  $u' = u - v$  and  $v' = u + v$ . Since  $v'$  does not occur in the quantum state, the sum over  $v'$  vanishes unless  $w_2 = w_1 + y' - y$ . Therefore, the state is

$$\sum_{\substack{(x,y,z),(x',y',z') \in H \\ u',w_1 \in \mathbb{Z}_p^n}} \omega_p^{\frac{1}{2}(-(y+y')u'+2(z'-z)+w_1(x+x'))} |u' + x - x', w_1\rangle \langle u', w_1 + y' - y|. \tag{16}$$

The diagonal entries are obtained by putting  $x = x'$  and  $y = y'$  and since  $|H| = |S_H|$ , we get  $z = z'$ . The diagonal entry is then proportional to

$$\sum_{\substack{(x,y,z) \in H \\ u',w_1 \in \mathbb{Z}_p^n}} \omega_p^{l(-yu'+w_1x)}. \tag{17}$$

Up to proportionality, this can be seen to be  $\chi_{w_1, -u'}(H)$ , a one dimensional irrep. Although the state is not diagonal after the Clebsch-Gordan transform, the diagonal entries correspond to one dimensional irreps.

## 5 The Quantum Algorithm

In this section, we present a quantum algorithm that operates on two copies of coset states at a time and show that it efficiently solves the HSP over  $G = \mathbb{Z}_p^{n+1} \rtimes \mathbb{Z}_p^n$ , where the input is  $n$  and  $\log p$ . The algorithm is as follows:

1. Obtain two copies of coset states for  $G$ .
2. Perform a quantum Fourier transform on each of the coset states and measure the irrep label and row index for each state. Assume that the measurement outcomes are high-dimensional irreps with labels  $k$  and  $l$ . With high probability the irreps are indeed both high dimensional and  $k+l \neq 0$ , when  $p > 2$  (see the analysis below). When  $p = 2$ , there is only one high dimensional irrep which occurs with probability  $1/2$  and  $k+l = 0$  always, since  $k = l = 1$ . We deal with this case at the end of this section. For now assume that  $p > 2$  and  $k+l \neq 0$ .
3. If  $-k/l$  is not a square in  $\mathbb{Z}_p$ , then we discard the pair  $(k, l)$  and obtain a new sample. Otherwise, perform a unitary  $U_\alpha \otimes I : |u, v\rangle \rightarrow |\alpha u, v\rangle$ , where  $\alpha$  is determined by the two irrep labels as  $\alpha = \sqrt{-k/l}$ . This leads to a “change” in the irrep label of the first state from  $k$  to  $-l$ . We can then apply the Clebsch-Gordan transform and obtain one dimensional irreps.
4. Apply a Clebsch-Gordan transform defined as

$$U_{CG} : |u, v\rangle \rightarrow \sum_{w \in \mathbb{Z}_p^n} \omega_p^{\frac{1}{2}(u+v)w} |u - v, w\rangle \tag{18}$$

to these states.

5. Measure the two registers in the standard basis. With the measurement outcomes, we have to perform some classical post-processing which involves finding the orthogonal space of a vector space.

Now, we present the analysis of the algorithm.

1. In step 1, we prepared the state  $\frac{1}{|G|} \sum_g |g\rangle|0\rangle$  and apply the black box  $U_f$  to obtain the state  $\frac{1}{|G|} \sum_g |g\rangle|f(g)\rangle$ . After discarding the second register, the resulting state is  $\frac{|H|}{|G|} |gH\rangle\langle gH|$ . We have two such copies.
2. After performing a QFT over  $G$  on two such copies, we measure the irrep label and a row index. The probability of measuring an irrep label  $\mu$  is given by  $p(\mu) = d_\mu \chi_\mu(H) |H|/|G|$ , where  $\chi_\mu$  is the character of the irrep. If  $\mu$  is a one-dimensional irrep, then the character is either 0 or 1 and so the probability becomes 0 or  $|H|/|G|$  accordingly. The character  $\chi_\mu(H) = 0$  if and only if  $\mu = (a, b) \in S_H^\perp$ . Therefore, the total probability of obtaining a one dimensional irrep is  $|H| |S_H^\perp|/|G|$ . Now, we have that  $|H| = |S_H|$  and so  $|H| |S_H^\perp| = p^{2n}$  since  $S_H^\perp$  is the orthogonal space in  $\mathbb{Z}_p^{2n}$ . Therefore, the total probability of obtaining a one dimensional irrep in the measurement is  $p^{2n}/p^{2n+1} = 1/p$ . This is exponentially small in the input size ( $\log p$ ). Therefore, the higher dimensional irreps occur with total probability of  $1 - 1/p$ . Since all of them have the same  $\chi_\mu(H) = p^n/|H|$ , each of them occurs with the same probability of  $1/p$ . Take two copies of coset states and perform weak Fourier sampling and obtain two high dimensional irreps  $k$  and  $l$ . The state is then  $\frac{|H|^2}{p^{2n}} \rho_k(H) \otimes \rho_l(H)$ . In the rest, we omit the normalization  $\frac{|H|}{p^n}$  of each register. Therefore, the state is proportional to

$$\rho_k(H) \otimes \rho_l(H) = \sum_{(x,y,z),(x',y',z') \in H} \omega_p^{k(z+yu)+l(z'+y'v)} |u+x, v+y\rangle\langle u, v|. \quad (19)$$

We can assume that  $k$  and  $l$  are such that  $k+l \neq 0$  since this happens with probability  $(p-1)/p^2$ . Now, choose  $\alpha = \sqrt{\frac{-k}{l}}$ . Since the equation  $lx^2 + k = 0$  has at most two solutions for any  $k, l \in \mathbb{Z}_p$ , for any given  $k, l$  chosen uniformly there exist solutions of the equation  $lx^2 + k = 0$  with probability  $1/2$ . Perform a unitary  $U_\alpha : |u\rangle \rightarrow |\alpha u\rangle$  on the first copy. The first register becomes proportional to

$$\begin{aligned} U_\alpha \rho_k(H) U_\alpha^\dagger &= \sum_{(x,y,z) \in H} \omega_p^{k(z+yu)} |\alpha(u+x)\rangle\langle \alpha u| \\ &= \sum_{(x,y,z) \in H, u_1 \in \mathbb{Z}_p^n} \omega_p^{\frac{k}{\alpha^2}(z_1+y_1 u_1)} |u_1+x_1\rangle\langle u| \\ &= \rho_{\frac{k}{\alpha^2}}(\phi_\alpha(H)), \end{aligned} \quad (20)$$

where  $(x_1, y_1, z_1) = \phi_\alpha(x, y, z) = (\alpha x, \alpha y, \alpha^2 z)$  and  $u_1 = \alpha u$ . It can be seen easily that  $\phi_\alpha$  is an isomorphism of  $G$  for  $\alpha \neq 0$  and hence  $\phi_\alpha(H)$  is subgroup of  $G$ . In fact,  $\phi_\alpha(H)$  is a conjugate of  $H$  since  $S_{\phi_\alpha(H)} = S_H$  (since if  $(x, y) \in S_H$ , then so is every multiple of it i.e.,  $(\alpha x, \alpha y) \in S_H$ ). Thus, we have obtained an irrep state with a new irrep label over a different subgroup. But this new subgroup is related to the old one by a known transformation.

In choosing the value of  $\alpha$  as above, we ensure that  $k/\alpha^2 = -l$  and hence obtain one dimensional irreps in the Clebsch-Gordan decomposition.

3. Perform a Clebsch-Gordan transform  $U_{CG}$  on the two copies of the coset states, i.e., perform the unitary given by the action

$$U_{CG} : |u, v\rangle \longrightarrow \sum_{w \in \mathbb{Z}_p^n} \omega_p^{\frac{1}{2}(u+v)w} |u - v, w\rangle. \quad (21)$$

The initial state of the two copies is

$$\begin{aligned} & \rho_{-l}(\phi_\alpha(H)) \otimes \rho_l(H) \\ = & \sum_{\substack{(x_1, y_1, z_1) \in \phi_\alpha(H), (x', y', z') \in H \\ u, v \in \mathbb{Z}_p^n}} \omega_p^{-l(z_1 + y_1 u) + l(z' + y' v)} |u + x_1, v + x'\rangle \langle u, v|. \end{aligned}$$

The resulting state after the transform is

$$\begin{aligned} & \sum_{\substack{(x_1, y_1, z_1) \in \phi_\alpha(H), (x', y', z') \in H \\ u, v, w_1, w_2 \in \mathbb{Z}_p^n}} \omega_p^{-l(z_1 + y_1 u) + l(z' + y' v) + \frac{1}{2}(u+v)(w_1 - w_2) + (x_1 + x')w_1} \times \\ & |u - v + x_1 - x', w_1\rangle \langle u - v, w_2| \\ = & \sum_{\substack{(x_1, y_1, z_1) \in \phi_\alpha(H), (x', y', z') \in H \\ u', v', w_1, w_2 \in \mathbb{Z}_p^n}} \omega_p^{-l(z_1 + y_1 \frac{u'+v'}{2}) + l(z' + y' \frac{v'-u'}{2}) + \frac{1}{2}(v')(w_1 - w_2) + (x_1 + x')w_1} \times \\ & |u' + x_1 - x', w_1\rangle \langle u', w_2|, \end{aligned}$$

where  $u' = u - v$  and  $v' = u + v$ . Notice that  $v'$  occurs only in the phase and not in the quantum states. Therefore, collecting the terms with  $v'$  we get

$$\sum_{v'} \omega_p^{\frac{1}{2}(y' - y_1 + w_1 - w_2)v'}. \quad (22)$$

This term is non-zero only when  $y' - y_1 + w_1 - w_2 = 0$ . Hence  $w_2 = w_1 - (y_1 - y')$ . Substituting this back in the equation, we get

$$\sum_{\substack{(x_1, y_1, z_1) \in \phi_\alpha(H), (x', y', z') \in H \\ u', w_1 \in \mathbb{Z}_p^n}} \omega_p^{\frac{1}{2}[(x_1 + x')w_1 - (y_1 + y')u' - 2(z_1 - z')]} |u' + x_1 - x', w_1\rangle \langle u', w_1 - (y_1 - y')|.$$

Reusing the labels  $u$  and  $v$  by putting  $u = u'$  and  $v = w_1 - (y_1 - y')$ , we obtain

$$\sum_{\substack{(x_1, y_1, z_1) \in \phi_\alpha(H), (x', y', z') \in H \\ u, v \in \mathbb{Z}_p^n}} \omega_p^{\frac{1}{2}[(x_1 + x')(v + (y_1 - y')) - (y_1 + y')u - 2(z_1 - z')]} |u + x_1 - x', v + y_1 - y'\rangle \langle u, v|.$$

This can be written as

$$\sum_{\substack{(x_1, y_1, z_1) \in \phi_\alpha(H), (x', y', z') \in H \\ u, v \in \mathbb{Z}_p^n}} \omega_p^{\frac{1}{2}[(x_1 + x')v - (y_1 + y')u - 2(z_1 - \frac{x_1 y_1}{2}) + 2(z' - \frac{x' y'}{2})]} |u + x_1 - x', v + y_1 - y'\rangle \langle u, v|.$$



$$\begin{aligned}
 (u'_2, v'_2) &= \left( \frac{u_2}{(1-\alpha_2)} - \frac{u_{n+1}}{(1-\alpha_{n+1})}, \frac{v_2}{(1-\alpha_2)} - \frac{v_{n+1}}{(1-\alpha_{n+1})} \right), \\
 &\quad \vdots \\
 (u'_n, v'_n) &= \left( \frac{u_n}{(1-\alpha_n)} - \frac{u_{n+1}}{(1-\alpha_{n+1})}, \frac{v_n}{(1-\alpha_n)} - \frac{v_{n+1}}{(1-\alpha_{n+1})} \right).
 \end{aligned}$$

With high probability, these vectors form a basis for  $S_H^\perp$  and hence we can determine  $S_H$  efficiently. This implies that the conjugacy class and hence the subgroup  $H_0$  is known. It remains only to determine  $(\hat{x}, \hat{y})$ . We can set  $(\hat{x}, \hat{y}) = (1 - \alpha_1)^{-1}(u_1 - u'_1, v_1 - v'_1)$  since the conjugating element can be determined up to addition by an element of  $S_H^\perp$ .  $H$  can be obtained with the knowledge of  $H_0$  and  $(\hat{x}, \hat{y})$ .

Finally, for completeness we consider the case  $p = 2$ . Assume that after Fourier sampling we have two high dimensional irreps with states given by

$$\rho_1(H) \otimes \rho_1(H) = \sum_{(x,y,z), (x',y',z') \in H, u,v \in \mathbb{Z}_2^n} (-1)^{z+z'+yu+y'v} |u+x, v+x'\rangle \langle u, v|. \tag{23}$$

The Clebsch-Gordan transform is given by the base change:

$$|u, v\rangle \rightarrow \sum_{w \in \mathbb{Z}_2^n} (-1)^{wv} |u+v, w\rangle. \tag{24}$$

Applying this to the two states, we obtain (in a similar manner as above)

$$\sum_{(x,y,z) \in H, u,v \in \mathbb{Z}_2^n} (-1)^{z+vx} \left( \sum_{(x',y',z') \in H} (-1)^{uy'+vx'} \right) |u+x, v+y\rangle \langle u, v|. \tag{25}$$

The inner sum is non-zero if and only if  $(u, v) \in S_H^\perp$ . Thus, measuring this state gives us  $S_H^\perp$  from which we can find  $S_H$ . We cannot determine  $H$  directly from here as in the case  $p > 2$ . But since we know  $S_H$ , we know the conjugacy class of  $H$  and we can determine the abelian group  $HG'$  which contains  $H$ . This group is obtained by appending the elements of  $S_H$  with every element of  $G' = \mathbb{Z}_2$  i.e., for  $(x, y) \in S_H$  we can say that  $(x, y, 0)$  and  $(x, y, 1)$  are in  $HG'$ . Once we know  $HG'$ , we now restrict the hiding function  $f$  to the abelian subgroup  $HG'$  of  $G$  and run the abelian version of the standard algorithm to find  $H$ .

## 6 Conclusions

Using the framework of coset states and non-abelian Fourier sampling we showed that the hidden subgroup problem for the Weyl-Heisenberg groups can be solved efficiently. In each iteration of the algorithm the quantum computer operates on  $k = 2$  coset states simultaneously which is an improvement over the previously best known quantum algorithm which required  $k = 4$  coset states. We believe

that the method of changing irrep labels and the technique of using Clebsch-Gordan transforms to devise multiregister experiments has some more potential for the solution of HSP over other groups. Finally, this group (at least when  $p = 2$ ) has importance in error correction. In fact, the state we obtain after Fourier sampling and measurement of an irrep is a projector onto the code space whose stabilizer generators are given by the generators of  $H$ . In view of this fact, it will be interesting to study the implications of the quantum algorithm derived in this paper to the design or decoding of quantum error-correcting codes.

## Acknowledgments

We thank Sean Hallgren and Pranab Sen for useful comments and discussions.

## References

- [AE07] Ambainis, A., Emerson, J.: Quantum  $t$ -designs:  $t$ -wise independence in the quantum world. In: Proceedings of the 22nd Annual IEEE Conference on Computational Complexity, pp. 129–140 (2007); Also arxiv preprint quant-ph/0701126
- [Bac08a] Bacon, D.: How a Clebsch-Gordan transform helps to solve the Heisenberg hidden subgroup problem. *Quantum Information and Computation* 8(5), 438–467 (2008)
- [Bac08b] Bacon, D.: Simon’s algorithm, Clebsch-Gordan sieves, and hidden symmetries of multiple squares (2008); Arxiv preprint quant-ph/0808.0174
- [BCD05] Bacon, D., Childs, A., van Dam, W.: From optimal measurement to efficient quantum algorithms for the hidden subgroup problem over semidirect product groups. In: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, pp. 469–478 (2005); Also arxiv preprint quant-ph/0504083
- [Bet87] Beth, T.: On the computational complexity of the general discrete Fourier transform. *Theoretical Computer Science* 51, 331–339 (1987)
- [BH97] Brassard, G., Høyer, P.: An exact polynomial-time algorithm for Simon’s problem. In: Proceedings of Fifth Israeli Symposium on Theory of Computing and Systems ISTCS, pp. 12–33. IEEE Computer Society Press, Los Alamitos (1997); Also arxiv preprint quant-ph/9704027
- [CRSS97] Calderbank, A.R., Rains, E.M., Shor, P.W., Sloane, N.J.A.: Quantum error correction and orthogonal geometry. *Physical Review Letters* 78(3), 405–408 (1997); Also arxiv preprint quant-ph/9605005
- [CRSS98] Calderbank, A.R., Rains, E.M., Shor, P.W., Sloane, N.J.A.: Quantum error correction via codes over  $GF(4)$ . *IEEE Transactions on Information Theory* 44(4), 1369–1387 (1998); Also arxiv preprint quant-ph/9608006
- [CSV07] Childs, A., Schulman, L.J., Vazirani, U.: Quantum algorithms for hidden nonlinear structures. In: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, pp. 395–404 (2007); Also preprint arxiv:0705.2784
- [DHI03] van Dam, W., Hallgren, S., Ip, L.: Quantum algorithms for some hidden shift problems. In: Proceedings of the Symposium on Discrete Algorithms (SODA), pp. 489–498 (2003); Also arxiv preprint quant-ph/0211140

- [EHK04] Ettinger, M., Høyer, P., Knill, E.: The quantum query complexity of the hidden subgroup problem is polynomial. *Information Processing Letters* 91(1), 43–48 (2004); Also arxiv preprint quant-ph/0401083
- [FIM+03] Friedl, K., Ivanyos, G., Magniez, F., Santha, M., Sen, P.: Hidden translation and orbit coset in quantum computing. In: *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, pp. 1–9 (2003); Also arxiv preprint quant-ph/0211091
- [Got96] Gottesman, D.: A class of quantum error-correcting codes saturating the quantum Hamming bound. *Physical Review A* 54(3), 1862–1868 (1996); Also arxiv preprint quant-ph/9604038
- [GSVV04] Grigni, M., Schulman, L., Vazirani, M., Vazirani, U.: Quantum mechanical algorithms for the nonabelian hidden subgroup problem. *Combinatorica*, 137–154 (2004)
- [Hal02] Hallgren, S.: Polynomial-time quantum algorithms for Pell’s equation and the principal ideal problem. In: *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pp. 653–658 (2002)
- [HH00] Hales, L., Hallgren, S.: An improved quantum Fourier transform algorithm and applications. In: *Proc. of the 41st Annual Symposium on Foundations of Computer Science (FOCS 2000)*, pp. 515–525. IEEE Computer Society, Los Alamitos (2000)
- [HMR+06] Hallgren, S., Moore, C., Rötteler, M., Russell, A., Sen, P.: Limitations of quantum coset states for graph isomorphism. In: *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pp. 604–617 (2006)
- [Høy97] Høyer, P.: Efficient Quantum Transforms (February 1997); Arxiv preprint quant-ph/9702028
- [HRT03] Hallgren, S., Russell, A., Ta-Shma, A.: The hidden subgroup problem and quantum computation using group representations. *SIAM Journal on Computing* 32(4), 916–934 (2003)
- [Hup83] Huppert, B.: *Endliche Gruppen*, vol. 1. Springer, Heidelberg (1983)
- [ISS07] Ivanyos, G., Sanselme, L., Santha, M.: An efficient algorithm for hidden subgroup problem in extraspecial groups. In: Thomas, W., Weil, P. (eds.) *STACS 2007*. LNCS, vol. 4393, pp. 586–597. Springer, Heidelberg (2007)
- [Kit97] Kitaev, A.Y.: Quantum computations: algorithms and error correction. *Russian Math. Surveys* 52(6), 1191–1249 (1997)
- [Kup05] Kuperberg, G.: A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM Journal on Computing* 35(1), 170–188 (2005); Also arxiv preprint quant-ph/0302112
- [ME98] Mosca, M., Ekert, A.: The hidden subgroup problem and eigenvalue estimation on a quantum computer. In: Williams, C.P. (ed.) *QCQC 1998*. LNCS, vol. 1509, pp. 174–188. Springer, Heidelberg (1999)
- [MRRS04] Moore, C., Rockmore, D., Russell, A., Schulman, L.: The power of basis selection in Fourier sampling: hidden subgroup problems in affine groups. In: *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1113–1122 (2004); Also arxiv preprint quant-ph/0503095
- [MZ04] Mosca, M., Zalka, C.H.: Exact quantum Fourier transforms and discrete logarithm algorithms. *International Journal of Quantum Information* 2(1), 91–100 (2004); Also arxiv preprint quant-ph/0301093
- [NC00] Nielsen, M., Chuang, I.: *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge (2000)
- [Reg04] Regev, R.: Quantum computation and lattice problems. *SIAM Journal on Computing* 33(3), 738–760 (2004)

- [RRS05] Radhakrishnan, J., Rötteler, M., Sen, P.: On the power of random bases in fourier sampling: Hidden subgroup problem in the heisenberg group. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 1399–1411. Springer, Heidelberg (2005)
- [Sen06] Sen, P.: Random measurement bases, quantum state distinction and applications to the hidden subgroup problem. In: Proceedings of the 21st Annual IEEE Conference on Computational Complexity, pp. 274–287 (2006); Also arxiv preprint quant-ph/0512085
- [Ser77] Serre, J.P.: Linear Representations of Finite Groups. Springer, Heidelberg (1977)
- [Sho97] Shor, P.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Journal on Computing 26(5), 1484–1509 (1997)

## A QFT for the Weyl-Heisenberg Groups

We briefly sketch how the quantum Fourier transform (QFT) can be computed for the Weyl-Heisenberg groups  $G_n = \mathbb{Z}_p^{n+1} \rtimes \mathbb{Z}_p^n$ . An implementation of the QFT for the case where  $p = 2$  was given in [Høy97]. This can be extended straightforwardly to  $p > 2$  as follows. Using Eq. (9), we obtain that the QFT for  $G_n$  is given by the unitary operator

$$\begin{aligned}
 \text{QFT}_{G_n} &= \sum_{a,b,x,y \in \mathbb{Z}_p^n, z \in \mathbb{Z}_p} \sqrt{\frac{1}{p^{2n+1}}} \omega_p^{ax+by} |0, a, b\rangle \langle z, x, y| \\
 &\quad + \sum_{\substack{a,b,x,y \in \mathbb{Z}_p^n \\ k \in \mathbb{Z}_p^*, z \in \mathbb{Z}_p}} \sqrt{\frac{p^n}{p^{2n+1}}} \omega_p^{k(z+by)} \delta_{x,a-b} |k, a, b\rangle \langle z, x, y| \\
 &= \sum_{\substack{a',b',x',y' \in \mathbb{Z}_p^{n-1} \\ a_n,b_n,x_n,y_n,z \in \mathbb{Z}_p}} \sqrt{\frac{1}{p^{2n-1}}} \frac{1}{p} \omega_p^{a'x'+b'y'} \omega_p^{a_n x_n + b_n y_n} \\
 &\quad |0, a' a_n, b' b_n\rangle \langle z, x' x_n, y' y_n| \\
 &\quad + \sum_{\substack{k \in \mathbb{Z}_p^*, a',b',x',y' \in \mathbb{Z}_p^{n-1} \\ z,a_n,b_n,x_n,y_n \in \mathbb{Z}_p}} \sqrt{\frac{p^{n-1}}{p^{2n-1}}} \frac{1}{\sqrt{p}} \omega_p^{k(z+b'y')} \omega_p^{k y_n b_n} \delta_{x',a'-b'} \delta_{x_n,a_n-b_n} \\
 &\quad |k, a' a_n, b' b_n\rangle \langle z, x' x_n, y' y_n| \\
 &= U \cdot \text{QFT}_{G_{n-1}}. \tag{26}
 \end{aligned}$$

The matrix  $U$  is given by

$$\begin{aligned}
 U &= \sum_{x_n, y_n, a_n, b_n \in \mathbb{Z}_p} \frac{1}{p} \omega_p^{a_n x_n + b_n y_n} |0\rangle \langle 0| \otimes |a_n, b_n\rangle \langle x_n, y_n| \\
 &\quad + \sum_{x_n, y_n, a_n, b_n \in \mathbb{Z}_p, k \in \mathbb{Z}_p^*} \frac{1}{\sqrt{p}} \omega_p^{b_n y_n} \delta_{x_n, a_n - b_n} |k\rangle \langle k| \otimes |a_n, b_n\rangle \langle x_n, y_n|
 \end{aligned}$$

$$= |0\rangle\langle 0| \otimes \text{QFT}_{\mathbb{Z}_p} \otimes \text{QFT}_{\mathbb{Z}_p} + \sum_{k \in \mathbb{Z}_p^*} V \cdot (I_p \otimes \text{QFT}_{\mathbb{Z}_p}^{(k)}), \tag{27}$$

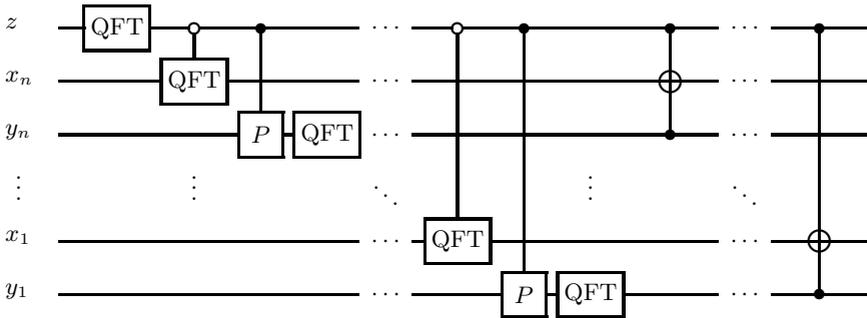
where  $I_p$  is the  $p$  dimensional identity matrix,

$$V = \sum_{u,v \in \mathbb{Z}_p} |u + v, v\rangle\langle u, v|, \tag{28}$$

and

$$\text{QFT}_{\mathbb{Z}_p}^{(k)} = \frac{1}{\sqrt{p}} \sum_{u,v \in \mathbb{Z}_p} \omega_p^{kuv} |u\rangle\langle v|. \tag{29}$$

From Eq. (27) and recursive application of Eq. (26) we obtain the efficient quantum circuit implementing  $\text{QFT}_{G_n}$  shown in Figure 1.



**Fig. 1.** QFT for the Weyl-Heisenberg group. The QFT gates shown in the circuit are QFTs for the cyclic groups  $\mathbb{Z}_p$ . Each of these QFTs can be implemented approximately [Kit97, HH00] or exactly [MZ04], in both cases with a complexity bounded by  $O(\log^2 p)$ . It should be noted that the wires in this circuit are actually  $p$ -dimensional systems. The meaning of the controlled gates where the control wire is an open circle is that the operation is applied to the target wire if and only if the control wire is in the state  $|0\rangle$ . The meaning of the controlled  $P$  gates where the control wire is a closed circle here means that the gate  $P_k$  is applied in case the control wire is in state  $|k\rangle$  with  $k \neq 0$ , and  $P_0 = I_p$ . Here  $P_k$  is the permutation matrix for which  $\text{QFT}^{(k)} = P_k \text{QFT}$  holds. The complexity of this circuit can be bounded by  $O(n \log^2 p)$ .

# Computing Equiangular Lines in Complex Space

Markus Grassl

Institute for Quantum Optics and Quantum Information  
Austrian Academy of Sciences, Innsbruck, Austria  
Markus.Grassl@oeaw.ac.at

**Abstract.** We consider the problem of finding equiangular lines in complex space, i. e., sets of unit vectors such that the modulus of the inner product between any two vectors is constant. We focus on the case of  $d^2$  such vectors in a space of dimension  $d$  which corresponds to so-called SIC-POVMs. We discuss how symmetries can be used to simplify the problem and how the corresponding system of polynomial equations can be solved using techniques of modular computation.

## 1 Introduction

The problem of finding equiangular lines has mainly been studied in Euclidean space (see, e. g., [15]), while much less is known for complex space. Equiangular lines in complex space are given by a set  $\{\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(m)}\}$  of  $m$  unit vectors in  $\mathbb{C}^m$  such that the modulus of the inner product between any two vectors is constant, i. e.,

$$|\langle \mathbf{v}^{(i)} | \mathbf{v}^{(j)} \rangle|^2 = \begin{cases} 1 & \text{for } i = j, \\ c & \text{for } i \neq j. \end{cases} \quad (1)$$

Here we focus on the case of  $d^2$  equiangular vectors, so-called SIC-POVMs, for which the condition reads

$$|\langle \mathbf{v}^{(i)} | \mathbf{v}^{(j)} \rangle|^2 = \begin{cases} 1 & \text{for } i = j, \\ \frac{1}{d+1} & \text{for } i \neq j. \end{cases} \quad (2)$$

These sets of equiangular lines have applications in different areas, for example, they give rise to an optimal POVM for quantum state tomography [9]. In his thesis, Zauner [18] provides algebraic solutions for  $d = 2, 3, 4, 5$ , and numerical solutions for  $d = 6$  and  $d = 7$ . An algebraic solution in dimension  $d = 8$  has been constructed by Hoggar [13, Example 8]. An explicit expression can be found in [18]. Using connections to the theory of tight frames, numerical solutions for (2) up to dimension 45 have been found [16]. It has been conjectured that SIC-POVMs exists for all dimensions [18,16,1]. Additional algebraic solutions for  $d = 7$  and  $d = 19$  have been found by Appleby [1]. His construction has been further investigated by Khatirinejad [14] who showed that it is unlikely that the construction based on the Legendre symbol can be generalized to other prime dimensions. The case of SIC-POVMs in prime dimension has also been studied

by Flammia [8]. The first solution for a non prime-power dimension,  $d = 6$ , was given by Grassl [10]. Solutions for dimensions  $d = 8, 9, 10, 11, 12, 13$ , and  $15$  are discussed in [11,12].

Here we mainly focus on the techniques that can be used for computing equiangular lines. We illustrate them constructing a new solution for  $d = 12$ .

## 2 The General Case

In the most general case, we would directly try to solve eq. (1) for a set of  $m$  vectors in  $\mathbb{C}^d$ . In order to be able to express the inner product and squared modulus as polynomials, we use  $2d$  real variables per vector, i. e.,

$$\mathbf{v}^{(j)} = (a_1^{(j)} + ib_1^{(j)}, \dots, a_d^{(j)} + ib_d^{(j)}),$$

where  $i^2 = -1$ . For  $d = 2$  and  $m = d^2 = 4$ , we get the following vectors:

$$\begin{aligned} \mathbf{v}^{(1)} &= (a_1^{(1)} + ib_1^{(1)}, a_2^{(1)} + ib_2^{(1)}) \\ \mathbf{v}^{(2)} &= (a_1^{(2)} + ib_1^{(2)}, a_2^{(2)} + ib_2^{(2)}) \\ \mathbf{v}^{(3)} &= (a_1^{(3)} + ib_1^{(3)}, a_2^{(3)} + ib_2^{(3)}) \\ \mathbf{v}^{(4)} &= (a_1^{(4)} + ib_1^{(4)}, a_2^{(4)} + ib_2^{(4)}) \end{aligned}$$

Since a unitary change of basis does not change the inner product between the vectors, without loss of generality, we can choose  $\mathbf{v}^{(1)} = (1, 0)$ . Furthermore, we can multiply any vector by a complex value of modulus one such that the first coordinate is real and positive, i. e.,  $b_1^{(j)} = 0$  and  $a_1^{(j)} = 1/\sqrt{c}$ . Finally, we can multiply the second coordinate of all vectors by the same complex value of modulus one such that the second coordinate of  $\mathbf{v}^{(2)}$  is real, i. e.,  $b_2^{(2)} = 0$ . While it is not difficult to solve the resulting system of polynomial equations for  $d = 2$ , already the case  $d = 3$  becomes hard for  $m > 4$ . In order to simplify the problem, we impose some symmetry.

## 3 Weyl-Heisenberg Symmetry

It has been conjectured that SIC-POVMs which are group-covariant with respect to the Weyl-Heisenberg group exist in any dimension [18,16]. The Weyl-Heisenberg group  $H_d$  is a finite subgroup with  $d^3$  elements of the group of unitary matrices  $U(d)$ . The group  $H_d$  is generated by the following two operators

$$X := \sum_{j=0}^{d-1} |j+1\rangle\langle j| \quad \text{and} \quad Z := \sum_{j=0}^{d-1} \omega_d^j |j\rangle\langle j|, \quad (3)$$

where  $\omega_d := \exp(2\pi i/d)$  is a primitive complex  $d$ -th root of unity and the cyclic shift  $X$  is modulo  $d$ . In (3) we have used bra-ket notation from quantum information where  $|j\rangle$  denotes the  $j$ -th basis vector as a column vector. The corresponding row vector is denoted by  $\langle j|$ . Each element of  $H_d$  can be uniquely

written as  $\omega_d^c X^a Z^b$  with  $a, b, c \in \{0, \dots, d - 1\}$ . Two elements  $\omega_d^c X^a Z^b$  and  $\omega_d^{c'} X^{a'} Z^{b'}$  commute iff  $ab' - a'b \equiv 0 \pmod d$ . The center  $\zeta(H_d)$  of the group  $H_d$  is generated by  $\omega_d I$ , where  $I$  denotes the identity matrix. Ignoring the global phase factor corresponding to the center  $\zeta(H_d)$ , the group  $H_d$  is isomorphic to the direct product of two cyclic groups of order  $d$ , i. e.,  $H_d/\zeta(H_d) \cong \mathbb{Z}_d \times \mathbb{Z}_d$  where  $\mathbb{Z}_d := \mathbb{Z}/d\mathbb{Z}$  denotes the ring of integers modulo  $d$ . The matrices  $X^a Z^b$  are mutually orthogonal with respect to the trace inner product and form a vector space basis of all  $d \times d$  matrices.

The ansatz is now to construct a SIC-POVM that is the orbit under  $H_d$ , i. e.,

$$\mathbf{v}^{(a,b)} := X^a Z^b \mathbf{v}^{(0)} \quad \text{for } a, b = 0, \dots, d - 1$$

$$\text{and } |\langle \mathbf{v}^{(a,b)} | \mathbf{v}^{(a',b')} \rangle|^2 = \begin{cases} 1 & \text{for } (a,b) = (a',b'), \\ 1/(d+1) & \text{for } (a,b) \neq (a',b'), \end{cases} \quad (4)$$

where we again use  $2d$  real variables  $x_0, \dots, x_{2d-1}$  for the fiducial vector  $\mathbf{v}^{(0)} = (x_0 + ix_1, \dots, x_{2j} + ix_{2j+1}, \dots, x_{2d-2} + ix_{2d-1})$ . Imposing the symmetry reduces the number of variables from  $O(d^3)$  to  $O(d)$ . The inner product does not change if we multiply  $\mathbf{v}^{(0)}$  by a complex number of modulus one. Hence, without loss of generality, we can choose the first coordinate to be real, i. e.,  $x_1 = 0$ . From (4) we get a system of polynomial equations for the  $2d - 1$  variables which we can solve using MAGMA [3] for dimension  $d = 3$  and  $d = 4$ .

But already the case  $d = 5$  gets rather complicated. Therefore we will use some additional symmetry to further simplify the problem.

### 4 Zauner's Conjecture

Based on the results for small dimension, Zauner [18] has conjectured that one can always find a SIC-POVM that possesses an additional symmetry of order three that stabilizes the fiducial vector  $\mathbf{v}^{(0)}$ . This additional symmetry is a particular element of the normalizer of  $H_d$  in the full unitary group, which is known as the Jacobi group  $J_d$  [10,11] or Clifford group [1]. The Clifford group is generated by the Fourier matrix

$$DF T_d = \frac{1}{\sqrt{d}} \sum_{j,k=0}^{d-1} \omega_d^{jk} |j\rangle \langle k|,$$

and a diagonal matrix  $P_d$  which is defined as follows:

$$P_d = \sum_{j=0}^{d-1} \omega_d^{j^2/2} |j\rangle \langle j| \quad \text{for } d \text{ even,}$$

$$P_d = \sum_{j=0}^{d-1} \omega_d^{j(j-1)/2} |j\rangle \langle j| \quad \text{for } d \text{ odd.}$$

The action of  $J_d$  on  $H_d$  modulo the center via conjugation is isomorphic to  $\text{SL}(2, \mathbb{Z}_d)$ , the group of  $2 \times 2$  matrices over the integers modulo  $d$  with unit determinant. Appleby [1] has verified that indeed all numerical solutions reported in [16] have a symmetry that is conjugated to the following element of the Clifford group given by Zauner [18, Section 3.4]:

$$S = \alpha \sum_{r,s=0}^{d-1} \omega_{2d}^{2rs+(d+1)s^2} |r\rangle\langle s|,$$

where  $\alpha$  is a normalization factor such that  $S^3 = I$ . Based on his analysis of the numerical solutions, Appleby conjectured that any SIC-POVM that is covariant with respect to the Weyl-Heisenberg group would possess an additional symmetry that is conjugated to Zauner's element  $S$ . However, the solution for  $d = 12$  given in [11] has an order-three symmetry that is not conjugated to Zauner's element. While this provides a counter-example to Appleby's strongest conjecture [1, Conjecture C], it remains open whether Zauner's conjecture is true that we can always find a solution starting with an eigenvector of  $S$ .

Compared to the ansatz in (4), we replace the general fiducial vector by a generic vector in one of the three eigenspaces of  $S$ . Zauner has shown that the dimension of these eigenspaces is  $m - 1$ ,  $m$ , or  $m + 1$ , where  $m = \lfloor d/3 \rfloor$  [18]. Hence the number of variables is reduced by a factor of approximately three. For  $d = 12$  we get a system of 15 polynomial equations in 9 variables over the cyclotomic field  $\mathbb{Q}(\omega_{24})$ . We did not succeed in directly solving this system of polynomial equations using the built-in functions of a computer algebra system such as MAGMA [3]. One of the reasons for this is that the coefficients of the polynomials in the initial system of equations have up to 40 digits.

## 5 Modular Techniques for Solving Polynomial Equations

One way of solving systems of polynomial equations is to use Gröbner bases [5], despite the fact that the running time of an algorithm for computing a Gröbner basis is in most of the cases exponential. The computer algebra system MAGMA [3] provides both a version of Buchberger's algorithm [4] and the so-called F4 algorithm of Faugère [6]. The latter uses linear algebra for the reduction of the polynomials, which is in many cases faster, but uses more memory. For polynomial rings over the rationals, modular techniques can be used for the linear algebra step, i.e., computation modulo several primes and recombining the results. This technique can also be applied when the polynomials are defined over a number field  $\mathbb{Q}(\theta)$ . For this, the element  $\theta$  is replaced by a new variable  $y$  and the minimal polynomial  $f(y)$  of  $\theta$  is added to the equations. The algorithm implemented in MAGMA uses this substitution by default.

As already mention, we have not been able to compute a Gröbner basis of our polynomials using the built-in algorithms of MAGMA. It turned out that the intermediate results have very large coefficients so that many primes have to be used. Already the second step of the F4 algorithm uses more than 300 primes, and the coefficients have more than 2200 digits.

A different approach to use modular algorithms for computing Gröbner bases is presented in [2]. The idea is to compute a Gröbner basis (GB) over prime fields  $\mathbb{Z}/p\mathbb{Z}$  and lift the result to  $\mathbb{Q}$ . For this, one can use a single prime number  $p$  and  $p$ -adic lifting, or many primes and Chinese remainder lifting [2, Section 6]. The disadvantage of the  $p$ -adic lifting is that in addition to the GB, the change of basis for the corresponding ideal has to be computed, i. e., the polynomials of the GB must be expressed in terms of the original generators. Another problem that applies to both techniques is that for some primes the result of the modular computation does not equal the basis that is obtained by reducing the coefficients of the GB over  $\mathbb{Q}$  modulo  $p$ , e. g., if  $p$  divides one of the denominators. However, it can be shown that for a given ideal, only a finite number of primes yield an incorrect result. The criteria given in [2] allow to decide for any pair of primes whether one of them is bad. Unfortunately, the computational complexity for checking these criteria is high as the Hilbert series of the ideal has to be computed. For practical purposes it turns out that we do not need to check the criteria, in particular, if we are only interested in solving a system of equations. In this case, we can always check whether a solution of the final system is also a solution for the initial system.

For our system, the computation of a single modular Gröbner basis using MAGMA V.2-14 on AMD Opteron processors (2.4 to 2.8 GHz) took about 40 hours and almost 17 GB of memory. In total we have used more than 300 primes with about 23 bit each, so the computation time would have been more than one CPU year. While it was not possible to reconstruct the full GB using only a few primes, it turned out that we could reconstruct some of the polynomials. Adding these polynomials to the initial system of equations reduced the computing time for a single modular GB to less than 2 hours, then further to less than 30 minutes, and finally down to a few seconds. The final GB with respect to graded reverse lexicographic (grevlex) order has 349 polynomials. Only 10 polynomials have coefficients whose numerator and denominator have less than 100 digits, the maximal numerators/denominators have almost 1000 digits.

The strategy is summarized as follows:

1. Let  $\mathcal{F} = \{f_1, \dots, f_m\} \subset \mathbb{Q}[x]$  denote the polynomials from the system of equations  $f_i = 0$  and set  $\mathcal{B} = \{ \}$ ,  $\ell = 1$ .
2. In step  $\ell$ , choose a new prime number  $p_\ell$  and compute a reduced Gröbner basis  $\mathcal{G}_{p_\ell} = \{g_1^{(p_\ell)}, \dots, g_m^{(p_\ell)}\}$  of the ideal generated by  $(\mathcal{F} \cup \mathcal{B})_{p_\ell} \subset (\mathbb{Z}/p_\ell\mathbb{Z})[x]$ . Here  $(\mathcal{S})_{p_\ell}$  denotes the set of polynomials obtained by reducing the coefficients of the polynomials in  $\mathcal{S} \subset \mathbb{Q}[x]$  modulo  $p_\ell$ .
3. Compute the polynomials  $\mathcal{G}_N = \{g_1^{(N)}, \dots, g_m^{(N)}\} \subset (\mathbb{Z}/N\mathbb{Z})[x]$  where  $N = p_1 \cdots p_\ell$  using the Chinese remainder theorem. A necessary condition to avoid bad prime numbers is that all modular Gröbner bases  $\mathcal{G}_{p_\ell}$  must have the same number of polynomials, and the leading monomials of the corresponding polynomials must be equal. If this is not the case, restart using different prime numbers.
4. Try to reconstruct the polynomials  $g_i \in \mathbb{Q}[x]$  from  $g_i^{(N)} \in (\mathbb{Z}/N\mathbb{Z})[x]$ . For this we have to invert the map  $\psi_N: \mathbb{Q} \rightarrow \mathbb{Z}/N\mathbb{Z}$  for all coefficients. The

command `RationalReconstruction` in `MAGMA` checks whether  $a \in \mathbb{Z}/N\mathbb{Z}$  has a unique pre-image  $\psi_N^{-1}(a) \in \mathbb{Q}$  and computes it (see, e.g., [17]). Add all polynomials  $g_i \in \mathbb{Q}[x]$  that have a unique reconstruction to the set  $\mathcal{B}$ .

5. While there is a polynomial  $g_i^{(N)}$  that cannot be reconstructed, increase  $\ell$  and goto Step 1.

In the next step, a Gröbner basis with respect to lexicographic order is computed using the FGLM algorithm [7] of `MAGMA` in about 4 hours. The final Gröbner basis contains 10 polynomials, some of which have coefficients with more than 18000 digits. The polynomials are of the form

$$\begin{array}{rcl}
 x_1 + & & G_1(x_9, x_{10}) = F_1 \\
 x_2 + & & G_2(x_9, x_{10}) = F_2 \\
 x_3 + & & G_3(x_9, x_{10}) = F_3 \\
 x_4 + & & G_4(x_9, x_{10}) = F_4 \\
 x_5 + & & G_5(x_9, x_{10}) = F_5 \\
 x_6 + & & G_6(x_9, x_{10}) = F_6 \\
 x_7^2 + x_7 H_7(x_9, x_{10}) + & & G_7(x_9, x_{10}) = F_7 \\
 x_8 + & & G_8(x_9, x_{10}) = F_8 \\
 x_9^{144} + & & G_9(x_9, x_{10}) = F_9 \\
 x_{10}^4 - 4x_{10}^2 + 1 = & & F_{10},
 \end{array}$$

where the degree of the polynomials  $G_i(x_9, x_{10})$  and  $H_7(x_9, x_{10})$  in the variables  $x_9$  and  $x_{10}$  is at most 143 and 3, respectively. The polynomial  $F_{10}$  is the minimal polynomial of  $\theta_{24} = \omega_{24} + 1/\omega_{24}$ , generating the real subfield  $\mathbb{Q}(\theta_{24})$  of the cyclotomic field  $\mathbb{Q}(\omega_{24})$ . Over  $\mathbb{Q}(\theta_{24})$ , the polynomial  $F_9(x_9, \theta_{24})$  has six irreducible factors of degree 24. Using one of the factors to define the field extension  $\mathbb{Q}(\theta_{24}, \rho_9)$  we obtain a number field of absolute degree 96. In the next step we solve the quadratic equation  $F_7(x_7, \rho_9, \theta_{24})$ , yielding the number field  $\mathbb{Q}(\theta_{24}, \rho_9, \rho_7)$  of absolute degree 192. This field has signature  $(96, 48)$ , i. e., it has a real embedding. This is important as we are only interested in real solutions for the variables  $x_i$ , while the variety corresponding to the system of polynomial equations is defined over the algebraic closure, i. e., the complex field  $\mathbb{C}$ . In order to express the fiducial vector, we have to adjoin the imaginary unit, hence the solution is over the field  $\mathbb{K} = \mathbb{Q}(\theta_{24}, \rho_9, \rho_7, \sqrt{-1})$  of absolute degree 384. The minimal polynomial of a random primitive element of this field has coefficients whose numerators/denominators have several hundred digits.

In order to simplify the representation of the field and thereby the representation of the solution, we use `MAGMA` to find subfields of small degree. This yields the following representation of a number field that contains at least one solution for  $d = 12$ :

$$\mathbb{L} = \mathbb{Q}(\sqrt{2}, \sqrt{3}, \sqrt{13}, \theta_1, \sqrt{\sqrt{13}-1}, \sqrt{\sqrt{13}-3}, \sqrt{-1}). \tag{5}$$

Here  $\Theta_1$  is a real solution of the equation  $\Theta_1^3 - 12\Theta_1 + 10 = 0$ . Note that the field  $\mathbb{L}$  in (5) has only degree 192, while the solution was found in the field  $\mathbb{K}$  of degree 384. This simplification is due to the fact that we consider a fiducial vector  $\mathbf{v}^{(0)}$  that is not normalized. The expressions for  $\mathbf{v}^{(0)}$  are given in the appendix.

Analyzing the resulting set of equiangular lines in dimension  $d = 12$  we find that it has indeed an additional symmetry of order three given by the matrix  $S$  of Zauner. As this is the only symmetry it follows that we obtain at least  $|\mathrm{SL}(2, \mathbb{Z}_{12})|/3 = 384$  different SIC-POVMs which have a symmetry of order three that is conjugated to the matrix  $S$ .

Note that the previous solution for dimension  $d = 12$  given in [11, Section 4.1] has also a symmetry of order three, but that symmetry  $T_{12}$  is not conjugated to the symmetry  $S$ . This follows from the fact that the multiplicities of the eigenvalues of  $T_{12}$  are 3, 3, and 6, whereas Zauner's matrix  $S$  has multiplicities 3, 4, and 5. Moreover, the fiducial vector given in [11, Table 2] has a much more compact representation.

## 6 Conclusions

Using modular techniques we have been able to compute a Gröbner basis for the system of polynomial equations corresponding to a SIC-POVM, i. e., a set of 144 equiangular lines in  $\mathbb{C}^{12}$  which have the matrix  $S$  of Zauner as additional symmetry. This provides additional support to Zauner's conjecture that this symmetry can be used to construct SIC-POVMs in any dimension. So far, the defining equations and the representation of the solutions are too complicated to derive a generalization to other dimensions. The next open case is  $d = 14$ , and we hope that the techniques illustrated here enable us to find a solution for this case as well.

## Acknowledgments

The author would like to thank the Institut für Algorithmen und Kognitive Systeme at Universität Karlsruhe (Germany) for kindly permitting to use their computer resources for the computations.

## References

1. Appleby, D.M.: SIC-POVMs and the Extended Clifford Group. *Journal of Mathematical Physics* 46, 052107 (2005); Preprint quant-ph/0412001
2. Arnold, E.A.: Modular Algorithms for Computing Gröbner Bases. *Journal of Symbolic Computation* 35(4), 403–419 (2003)
3. Bosma, W., Cannon, J.J., Playoust, C.: The Magma Algebra System I: The User Language. *Journal of Symbolic Computation* 24(3–4), 235–266 (1997)
4. Buchberger, B.: A Theoretical Basis for the Reduction of Polynomials to Canonical Forms. *ACM SIGSAM Bulletin* 10(3), 19–29 (1976)

5. Cox, D.A., Little, J.B., O’Shea, D.: Ideals, Varieties, and Algorithms. Springer, New York (1992)
6. Faugère, J.C.: A New Efficient Algorithm for Computing Gröbner Bases (F4). Journal of Pure and Applied Algebra 139(1–3), 61–88 (1999)
7. Faugère, J.C., Gianni, P., Lazard, D., Mora, T.: Efficient Computations of Zero-dimensional Gröbner Bases by Change of Ordering. Journal of Symbolic Computation 16(4), 329–344 (1993)
8. Flammia, S.T.: On SIC-POVMs in Prime Dimensions. Journal of Physics A 39(43), 13483–13493 (2006); Preprint quant-ph/0605050
9. Flammia, S.T., Silberfarb, A., Caves, C.M.: Minimal Informationally Complete Measurements for Pure States. Foundations of Physics 35(12), 1985–2006 (2005); Preprint quant-ph/0404137
10. Grassl, M.: On SIC-POVMs and MUBs in Dimension 6. In: Proceedings ERATO Conference on Quantum Information Science 2004 (EQIS 2004), Tokyo, pp. 60–61 (September 2004); Preprint quant-ph/0406175
11. Grassl, M.: Tomography of Quantum States in Small Dimensions. Electronic Notes in Discrete Mathematics 20, 151–164 (2005)
12. Grassl, M.: Finding Equiangular Lines in Complex Space. Talk at the MAGMA 2006 Conference, Technische Universität Berlin (July 2006)
13. Hoggar, S.G.:  $t$ -Designs in Projective Spaces. European Journal of Combinatorics 3, 233–254 (1982)
14. Khatirinejad, M.: On Weyl-Heisenberg Orbits of Equiangular Lines. Journal of Algebraic Combinatorics 28(3), 333–349 (2007) (Published online November 6, 2007)
15. Lemmens, P.H.W., Seidel, J.J.: Equiangular Lines. Journal of Algebra 24(3), 494–512 (1973)
16. Renes, J.M., Blume-Kohout, R., Scott, A.J., Caves, C.M.: Symmetric Informationally Complete Quantum Measurements. Journal of Mathematical Physics 45(6), 2171–2180 (2004); Preprint quant-ph/0310075
17. Wang, P.S., Guy, M.J.T., Davenport, J.H.:  $P$ -adic Reconstruction of Rational Numbers. ACM SIGSAM Bulletin 16(2), 2–3 (1982)
18. Zauner, G.: Quantendesigns: Grundzüge einer nichtkommutativen Designtheorie. Dissertation, Universität Wien (1999)

## Appendix

Below we list the components  $v_i$  of the non-normalized fiducial vector  $\mathbf{v}^{(0)} = (v_1, \dots, v_{12})$ , where  $\Theta_1$  is a real solution of  $\Theta_1^3 - 12\Theta_1 + 10 = 0$  and  $i^2 = -1$ .

$$v_1 = 61018151288640$$

$$v_2 = \left( \begin{aligned} & (16454168751\sqrt{312(4-\sqrt{13})} + 47390453990\sqrt{78(\sqrt{13}-1)} - 150615625836\sqrt{78(\sqrt{13}-3)} \\ & + 476309407899\sqrt{24(4-\sqrt{13})} + 178740295623\sqrt{104(4-\sqrt{13})} - 6550417786\sqrt{156(4-\sqrt{13})} - 200574075210\sqrt{78} \\ & + 2171251706\sqrt{6(\sqrt{13}-1)} + 7529062602\sqrt{26(\sqrt{13}-1)} + 57774655044\sqrt{39(\sqrt{13}-1)} \\ & - 214365468304\sqrt{6(\sqrt{13}-3)} - 296443549050\sqrt{26(\sqrt{13}-3)} - 292212076266\sqrt{39(\sqrt{13}-3)} \\ & + 1462364383779\sqrt{8(4-\sqrt{13})} - 32306974726\sqrt{12(4-\sqrt{13})} + 288116397432\sqrt{52(4-\sqrt{13})} - 920784383298\sqrt{6} \\ & - 65280647370\sqrt{26} - 236802250056\sqrt{39} + 18984505410\sqrt{2(\sqrt{13}-1)} - 89743882176\sqrt{3(\sqrt{13}-1)} \\ & + 64219197876\sqrt{13(\sqrt{13}-1)} - 1210153009962\sqrt{2(\sqrt{13}-3)} - 854131510830\sqrt{3(\sqrt{13}-3)} \\ & - 564912731370\sqrt{13(\sqrt{13}-3)} + 886418028756\sqrt{4(4-\sqrt{13})} - 3175132080354\sqrt{2} - 2214035946096\sqrt{3} \end{aligned} \right)$$

$$\begin{aligned}
& -782921827236\sqrt{13}-331481255040\sqrt{\sqrt{13}-1}-2116117476798\sqrt{\sqrt{13}-3}-1175644168764)\Theta_1^2 \\
& + (261285520014\sqrt{312(4-\sqrt{13})}+217409051218\sqrt{78(\sqrt{13}-1)}-622316394570\sqrt{78(\sqrt{13}-3)} \\
& +1400420673054\sqrt{24(4-\sqrt{13})}-77047066524\sqrt{104(4-\sqrt{13})}+633787226572\sqrt{156(4-\sqrt{13})} \\
& -611561015256\sqrt{78}-582751308854\sqrt{6(\sqrt{13}-1)}+11268257814\sqrt{26(\sqrt{13}-1)}+151941836640\sqrt{39(\sqrt{13}-1)} \\
& -1715193142130\sqrt{6(\sqrt{13}-3)}-455563942290\sqrt{26(\sqrt{13}-3)}-926400471348\sqrt{39(\sqrt{13}-3)} \\
& +1572951646968\sqrt{8(4-\sqrt{13})}+1182760486024\sqrt{12(4-\sqrt{13})}+491208218736\sqrt{52(4-\sqrt{13})} \\
& -2382877556280\sqrt{6}-525308225280\sqrt{26}-750058129428\sqrt{39}+127877818926\sqrt{2(\sqrt{13}-1)} \\
& -264092706384\sqrt{3(\sqrt{13}-1)}+100408016484\sqrt{13(\sqrt{13}-1)}-1885983791250\sqrt{2(\sqrt{13}-3)} \\
& -2558306038236\sqrt{3(\sqrt{13}-3)}-814865740884\sqrt{13(\sqrt{13}-3)}+152263837284\sqrt{4(4-\sqrt{13})}-5651397155352\sqrt{2} \\
& -2529045217452\sqrt{3}-2261705398908\sqrt{13}+1107438540\sqrt{\sqrt{13}-1}-2431312258596\sqrt{\sqrt{13}-3} \\
& -2394241466916)\Theta_1-76485755454\sqrt{312(4-\sqrt{13})}-18112893910\sqrt{78(\sqrt{13}-1)}+833223741870\sqrt{78(\sqrt{13}-3)} \\
& -419444816042\sqrt{24(4-\sqrt{13})}-2615948219322\sqrt{104(4-\sqrt{13})}-612061589596\sqrt{156(4-\sqrt{13})} \\
& +1194861591888\sqrt{78}-1790482702138\sqrt{6(\sqrt{13}-1)}-522547672758\sqrt{26(\sqrt{13}-1)} \\
& -856190279844\sqrt{39(\sqrt{13}-1)}+1475794800974\sqrt{6(\sqrt{13}-3)}+2058789844194\sqrt{26(\sqrt{13}-3)} \\
& +2449219436712\sqrt{39(\sqrt{13}-3)}-12685094068842\sqrt{8(4-\sqrt{13})}-1362249350176\sqrt{12(4-\sqrt{13})} \\
& -1458981200556\sqrt{52(4-\sqrt{13})}+3249228240648\sqrt{6}-115720418712\sqrt{26}+127070102184\sqrt{39} \\
& +1349248210830\sqrt{2(\sqrt{13}-1)}+2477933969940\sqrt{3(\sqrt{13}-1)}-9103835640\sqrt{13(\sqrt{13}-1)} \\
& +6757060637730\sqrt{2(\sqrt{13}-3)}+6316192474800\sqrt{3(\sqrt{13}-3)}+4731647487060\sqrt{13(\sqrt{13}-3)} \\
& -8516766878568\sqrt{4(4-\sqrt{13})}+13033338017544\sqrt{2}+24092955961152\sqrt{3}+3147389886912\sqrt{13} \\
& -1384757681112\sqrt{\sqrt{13}-1}+21043552558236\sqrt{\sqrt{13}-3}+3686540539032)\Theta_1+ (131613425423\sqrt{312(4-\sqrt{13})} \\
& -88551276122\sqrt{78(\sqrt{13}-1)}-142142839110\sqrt{78(\sqrt{13}-3)}+375887075759\sqrt{24(4-\sqrt{13})} \\
& +453389075337\sqrt{104(4-\sqrt{13})}+231627929636\sqrt{156(4-\sqrt{13})}+21828322638\sqrt{78}+481486852054\sqrt{6(\sqrt{13}-1)} \\
& -198169198230\sqrt{26(\sqrt{13}-1)}+126014805704\sqrt{39(\sqrt{13}-1)}-448454701006\sqrt{6(\sqrt{13}-3)} \\
& -581300086716\sqrt{26(\sqrt{13}-3)}+18771974994\sqrt{39(\sqrt{13}-3)}+1728489498177\sqrt{8(4-\sqrt{13})} \\
& +752422831472\sqrt{12(4-\sqrt{13})}+615432666246\sqrt{52(4-\sqrt{13})}+279497636574\sqrt{6}-351661391322\sqrt{26} \\
& +212102566188\sqrt{39}+1208292243366\sqrt{2(\sqrt{13}-1)}-209569880260\sqrt{3(\sqrt{13}-1)}+19112768400\sqrt{13(\sqrt{13}-1)} \\
& -1996399994160\sqrt{2(\sqrt{13}-3)}-759696906722\sqrt{3(\sqrt{13}-3)}-505441691178\sqrt{13(\sqrt{13}-3)} \\
& +2891717726586\sqrt{4(4-\sqrt{13})}+411912780474\sqrt{2}-63843827580\sqrt{3}+3662516664\sqrt{13} \\
& +701727865644\sqrt{\sqrt{13}-1}-3024753476550\sqrt{\sqrt{13}-3}+234414217440)\Theta_1^2+(485953705408\sqrt{312(4-\sqrt{13})} \\
& -77314369138\sqrt{78(\sqrt{13}-1)}-536153615922\sqrt{78(\sqrt{13}-3)}+1762174555492\sqrt{24(4-\sqrt{13})} \\
& +865570530858\sqrt{104(4-\sqrt{13})}+591990607756\sqrt{156(4-\sqrt{13})}-76032122904\sqrt{78}+975655405790\sqrt{6(\sqrt{13}-1)} \\
& -338796465690\sqrt{26(\sqrt{13}-1)}+553054838164\sqrt{39(\sqrt{13}-1)}-1142564272538\sqrt{6(\sqrt{13}-3)} \\
& -866657902062\sqrt{26(\sqrt{13}-3)}-467801756460\sqrt{39(\sqrt{13}-3)}+3797772147546\sqrt{8(4-\sqrt{13})} \\
& +2110836625768\sqrt{12(4-\sqrt{13})}+870804592512\sqrt{52(4-\sqrt{13})}+872674965552\sqrt{6}+1088385539448\sqrt{26} \\
& +831899830620\sqrt{39}+2039495559894\sqrt{2(\sqrt{13}-1)}-1137607262324\sqrt{3(\sqrt{13}-1)}+470657563224\sqrt{13(\sqrt{13}-1)} \\
& -1327088115054\sqrt{2(\sqrt{13}-3)}-3498722425900\sqrt{3(\sqrt{13}-3)}-1058047600620\sqrt{13(\sqrt{13}-3)} \\
& +4507866532572\sqrt{4(4-\sqrt{13})}-719522191128\sqrt{2}-304693140492\sqrt{3}-160741716228\sqrt{13} \\
& -808216646952\sqrt{\sqrt{13}-1}-4988505234180\sqrt{\sqrt{13}-3}+2487569153940)\Theta_1-656255268622\sqrt{312(4-\sqrt{13})} \\
& +351457959010\sqrt{78(\sqrt{13}-1)}+236345270010\sqrt{78(\sqrt{13}-3)}-726791099086\sqrt{24(4-\sqrt{13})} \\
& -3337508108166\sqrt{104(4-\sqrt{13})}-925162539748\sqrt{156(4-\sqrt{13})}-470321980896\sqrt{78} \\
& -2029375456394\sqrt{6(\sqrt{13}-1)}+1699183330206\sqrt{26(\sqrt{13}-1)}-198562829152\sqrt{39(\sqrt{13}-1)} \\
& +2426964564362\sqrt{6(\sqrt{13}-3)}+1152493002762\sqrt{26(\sqrt{13}-3)}-780968811684\sqrt{39(\sqrt{13}-3)} \\
& -13567726519386\sqrt{8(4-\sqrt{13})}-3503131757008\sqrt{12(4-\sqrt{13})}-986748236148\sqrt{52(4-\sqrt{13})} \\
& -1333270036176\sqrt{6}-3105402077496\sqrt{26}-2125863620448\sqrt{39}-9982957572510\sqrt{2(\sqrt{13}-1)} \\
& -36938218720\sqrt{3(\sqrt{13}-1)}+210082066860\sqrt{13(\sqrt{13}-1)}+2482683986250\sqrt{2(\sqrt{13}-3)} \\
& +796530472660\sqrt{3(\sqrt{13}-3)}+3521848188000\sqrt{13(\sqrt{13}-3)}-68975834433504\sqrt{4(4-\sqrt{13})}+9315916177152\sqrt{2} \\
& +3878857592376\sqrt{3}-862614219384\sqrt{13}-3150102399756\sqrt{\sqrt{13}-1}+23829222568008\sqrt{\sqrt{13}-3} \\
& +7780537270896
\end{aligned}$$

$$\begin{aligned}
v_3 = & \left( \left( 139285184364\sqrt{78(\sqrt{13}-1)} + 131930836488\sqrt{78} - 147485629512\sqrt{6(\sqrt{13}-1)} \right. \right. \\
& + 49195045740\sqrt{26(\sqrt{13}-1)} + 16613832912\sqrt{39(\sqrt{13}-1)} + 976183494312\sqrt{6} - 295192621944\sqrt{26} \\
& - 68424768396\sqrt{39} + 204107810712\sqrt{2(\sqrt{13}-1)} + 393914221584\sqrt{3(\sqrt{13}-1)} - 126420937752\sqrt{13(\sqrt{13}-1)} \\
& - 981357553800\sqrt{2} - 316722878940\sqrt{3} + 128818185732\sqrt{13} + 895795493736\sqrt{\sqrt{13}-1} - 1127988040140 \left. \right) \Theta_1^2 \\
& + \left( 569907388620\sqrt{78(\sqrt{13}-1)} + 309663347508\sqrt{78} - 1283601293208\sqrt{6(\sqrt{13}-1)} \right. \\
& + 296801047668\sqrt{26(\sqrt{13}-1)} + 292036518720\sqrt{39(\sqrt{13}-1)} - 926166269964\sqrt{6} - 877793935176\sqrt{26} \\
& - 894131968788\sqrt{39} - 275676785280\sqrt{2(\sqrt{13}-1)} + 128811390552\sqrt{3(\sqrt{13}-1)} - 227120191392\sqrt{13(\sqrt{13}-1)} \\
& + 2252150700696\sqrt{2} + 429274655004\sqrt{3} + 1747850662476\sqrt{13} + 2168480817360\sqrt{\sqrt{13}-1} - 2823852974436 \left. \right) \Theta_1 \\
& - 545489448408\sqrt{78(\sqrt{13}-1)} - 1120807155228\sqrt{78} - 569984826528\sqrt{6(\sqrt{13}-1)} - 422058557148\sqrt{26(\sqrt{13}-1)} \\
& - 522845214744\sqrt{39(\sqrt{13}-1)} - 1953716627148\sqrt{6} + 1601894333772\sqrt{26} + 810795451416\sqrt{39} \\
& - 918181829604\sqrt{2(\sqrt{13}-1)} - 1341924188592\sqrt{3(\sqrt{13}-1)} + 1167531821808\sqrt{13(\sqrt{13}-1)} \\
& + 17969533018476\sqrt{2} + 756364528888\sqrt{3} - 806813224272\sqrt{13} - 10018467042792\sqrt{\sqrt{13}-1} + 9332185881408 \left. \right) i \\
& + \left( -54431200792\sqrt{78(\sqrt{13}-1)} - 260181139440\sqrt{78} + 421573853012\sqrt{6(\sqrt{13}-1)} \right. \\
& - 220722412968\sqrt{26(\sqrt{13}-1)} - 9926924408\sqrt{39(\sqrt{13}-1)} + 1330427190144\sqrt{6} + 137956773480\sqrt{26} \\
& + 562840700988\sqrt{39} + 1724896803708\sqrt{2(\sqrt{13}-1)} + 269745720088\sqrt{3(\sqrt{13}-1)} - 186585492432\sqrt{13(\sqrt{13}-1)} \\
& - 1529086474968\sqrt{2} - 1611741636804\sqrt{3} - 432827297340\sqrt{13} + 1891035603600\sqrt{\sqrt{13}-1} + 2072219149572 \left. \right) \Theta_1^2 \\
& + \left( 123242131624\sqrt{78(\sqrt{13}-1)} - 629423531592\sqrt{78} + 538898127820\sqrt{6(\sqrt{13}-1)} - 153671692320\sqrt{26(\sqrt{13}-1)} \right. \\
& + 258331417808\sqrt{39(\sqrt{13}-1)} + 2483733732792\sqrt{6} + 510466739628\sqrt{26} + 1480412324364\sqrt{39} \\
& + 1634833517148\sqrt{2(\sqrt{13}-1)} + 420799452320\sqrt{3(\sqrt{13}-1)} + 120592839720\sqrt{13(\sqrt{13}-1)} - 288249093444\sqrt{2} \\
& - 3477795150804\sqrt{3} - 1441005667356\sqrt{13} + 1103401094016\sqrt{\sqrt{13}-1} + 1046895667428 \left. \right) \Theta_1 \\
& + 680271684356\sqrt{78(\sqrt{13}-1)} + 1535540157276\sqrt{78} - 3286811550724\sqrt{6(\sqrt{13}-1)} \\
& + 1808776281456\sqrt{26(\sqrt{13}-1)} + 171008023768\sqrt{39(\sqrt{13}-1)} + 952537513356\sqrt{6} - 2410441570884\sqrt{26} \\
& - 3236183383632\sqrt{39} - 10865629931832\sqrt{2(\sqrt{13}-1)} - 275830972976\sqrt{3(\sqrt{13}-1)} \\
& + 2431813069824\sqrt{13(\sqrt{13}-1)} + 28226661439548\sqrt{2} + 15801032567424\sqrt{3} + 4492371086184\sqrt{13} \\
& - 14482308183096\sqrt{\sqrt{13}-1} - 3903632907096
\end{aligned}$$

$$\begin{aligned}
v_4 = & \left( \left( -731915712552\sqrt{78(\sqrt{13}-3)} + 125519540184\sqrt{78} - 2210765462464\sqrt{6(\sqrt{13}-3)} \right. \right. \\
& - 722872066380\sqrt{26(\sqrt{13}-3)} - 797653767444\sqrt{39(\sqrt{13}-3)} + 458077199424\sqrt{6} + 323269220712\sqrt{26} \\
& - 366532023072\sqrt{39} - 2555517112980\sqrt{2(\sqrt{13}-3)} - 3878884987380\sqrt{3(\sqrt{13}-3)} - 508596806388\sqrt{13(\sqrt{13}-3)} \\
& - 108644753568\sqrt{2} - 197578285008\sqrt{3} - 100156084800\sqrt{13} - 4395208196964\sqrt{\sqrt{13}-3} + 911613168960 \left. \right) \Theta_1^2 \\
& + \left( -1488974296632\sqrt{78(\sqrt{13}-3)} + 252967696776\sqrt{78} - 3042281257184\sqrt{6(\sqrt{13}-3)} \right. \\
& - 2064024790056\sqrt{26(\sqrt{13}-3)} - 1984448071968\sqrt{39(\sqrt{13}-3)} - 1522918860216\sqrt{6} + 466687258656\sqrt{26} \\
& - 1039084227960\sqrt{39} - 5313676608864\sqrt{2(\sqrt{13}-3)} - 7546811272416\sqrt{3(\sqrt{13}-3)} \\
& - 2218271010264\sqrt{13(\sqrt{13}-3)} - 1373559746208\sqrt{2} + 3006982158648\sqrt{3} + 724293724632\sqrt{13} \\
& - 12927479536296\sqrt{\sqrt{13}-3} + 2976810231720 \left. \right) \Theta_1 + 1985716744632\sqrt{78(\sqrt{13}-3)} - 127930187208\sqrt{78} \\
& + 3958478787224\sqrt{6(\sqrt{13}-3)} + 2767825654224\sqrt{26(\sqrt{13}-3)} + 5971067624712\sqrt{39(\sqrt{13}-3)} \\
& + 2083313767704\sqrt{6} - 462949809576\sqrt{26} + 1710278238192\sqrt{39} + 14037954330816\sqrt{2(\sqrt{13}-3)} \\
& + 30145415042808\sqrt{3(\sqrt{13}-3)} + 2388741052008\sqrt{13(\sqrt{13}-3)} + 7723861751736\sqrt{2} + 15805866028368\sqrt{3} \\
& - 127993749792\sqrt{13} + 23433143976216\sqrt{\sqrt{13}-3} - 635011026624 \left. \right) i + \left( 154300709940\sqrt{78(\sqrt{13}-3)} \right. \\
& - 269227309776\sqrt{78} + 761698308956\sqrt{6(\sqrt{13}-3)} - 129453209208\sqrt{26(\sqrt{13}-3)} + 583684706364\sqrt{39(\sqrt{13}-3)} \\
& + 1490196729528\sqrt{6} - 270624017040\sqrt{26} + 411101580336\sqrt{39} - 1353303589248\sqrt{2(\sqrt{13}-3)} \\
& + 1356420570076\sqrt{3(\sqrt{13}-3)} + 371194537620\sqrt{13(\sqrt{13}-3)} - 1457459351400\sqrt{2} - 1424901505872\sqrt{3} \\
& - 327820027392\sqrt{13} - 462358944060\sqrt{\sqrt{13}-3} - 2700029672496 \left. \right) \Theta_1^2 + \left( -80589673632\sqrt{78(\sqrt{13}-3)} \right. \\
& - 1121899025424\sqrt{78} + 743419518712\sqrt{6(\sqrt{13}-3)} + 1000291281648\sqrt{26(\sqrt{13}-3)} \\
& + 347063984424\sqrt{39(\sqrt{13}-3)} + 1808068270320\sqrt{6} - 68981441880\sqrt{26} + 571067720616\sqrt{39} \\
& + 3818491311336\sqrt{2(\sqrt{13}-3)} - 1067410167304\sqrt{3(\sqrt{13}-3)} + 1721153813424\sqrt{13(\sqrt{13}-3)} \\
& + 1165371165192\sqrt{2} - 5157949405128\sqrt{3} - 229994431992\sqrt{13} + 2686412880528\sqrt{\sqrt{13}-3} - 3521174897448 \left. \right) \Theta_1 \\
& - 1822444574184\sqrt{78(\sqrt{13}-3)} + 952582707096\sqrt{78} - 4330096073368\sqrt{6(\sqrt{13}-3)}
\end{aligned}$$

$$\begin{aligned}
& -1347178222848\sqrt{26(\sqrt{13}-3)}-5512616298744\sqrt{39(\sqrt{13}-3)}-3348263555208\sqrt{6}-417961305528\sqrt{26} \\
& -1596753185184\sqrt{39}-1944700416672\sqrt{2(\sqrt{13}-3)}-20247022085576\sqrt{3(\sqrt{13}-3)} \\
& -3825810122136\sqrt{13(\sqrt{13}-3)}+8604302416008\sqrt{2}+8486483594784\sqrt{3}+4430219876784\sqrt{13} \\
& +4880645143608\sqrt{\sqrt{13}-3}+23270763245808
\end{aligned}$$

$$\begin{aligned}
v_5 = & \left( \left( 44504276384\sqrt{78(\sqrt{13}-1)}+318237636924\sqrt{78}-151828132924\sqrt{6(\sqrt{13}-1)} \right. \right. \\
& +34136920536\sqrt{26(\sqrt{13}-1)}-98935477176\sqrt{39(\sqrt{13}-1)}+242322009228\sqrt{6}+931345797708\sqrt{26} \\
& -186392234436\sqrt{39}+166138799892\sqrt{2(\sqrt{13}-1)}+573401985936\sqrt{3(\sqrt{13}-1)}-254859333504\sqrt{13(\sqrt{13}-1)} \\
& -58488656148\sqrt{2}+1175430629724\sqrt{3}-215870718492\sqrt{13}+1558758003816\sqrt{\sqrt{13}-1}+3336815059812 \Big) \theta_1^2 \\
& + \left( 135089286184\sqrt{78(\sqrt{13}-1)}+691888522860\sqrt{78}-118098675500\sqrt{6(\sqrt{13}-1)}+274264532040\sqrt{26(\sqrt{13}-1)} \right. \\
& -11847154560\sqrt{39(\sqrt{13}-1)}-1464891106068\sqrt{6}+2314315973448\sqrt{26}-175448592660\sqrt{39} \\
& -531432423132\sqrt{2(\sqrt{13}-1)}+656996803320\sqrt{3(\sqrt{13}-1)}-427936224360\sqrt{13(\sqrt{13}-1)}-3688428786120\sqrt{2} \\
& +5409811905420\sqrt{3}+69175002780\sqrt{13}+2166265940280\sqrt{\sqrt{13}-1}+10349226330300 \Big) \theta_1 \\
& -509263660588\sqrt{78(\sqrt{13}-1)}-938070648468\sqrt{78}+3010980577748\sqrt{6(\sqrt{13}-1)}+623036788368\sqrt{26(\sqrt{13}-1)} \\
& +1189535344944\sqrt{39(\sqrt{13}-1)}+7379540374284\sqrt{6}-3958070812236\sqrt{26}-1152333318408\sqrt{39} \\
& -3616678251264\sqrt{2(\sqrt{13}-1)}-6297792128472\sqrt{3(\sqrt{13}-1)}+1185739493088\sqrt{13(\sqrt{13}-1)} \\
& -9563947783884\sqrt{2}+5672324504808\sqrt{3}-1976154581232\sqrt{13}-7248951680568\sqrt{\sqrt{13}-1}-10695965318064 \Big) i \\
& + \left( 122671351452\sqrt{78(\sqrt{13}-1)}+142230668148\sqrt{78}-541399851096\sqrt{6(\sqrt{13}-1)}+175615983492\sqrt{26(\sqrt{13}-1)} \right. \\
& -261956535816\sqrt{39(\sqrt{13}-1)}+2240309341764\sqrt{6}+118086900084\sqrt{26}+336823402572\sqrt{39} \\
& -691687683024\sqrt{2(\sqrt{13}-1)}+688885480608\sqrt{3(\sqrt{13}-1)}-224811029232\sqrt{13(\sqrt{13}-1)}+2371273089588\sqrt{2} \\
& +1274473966428\sqrt{3}+212079746988\sqrt{13}+487579872312\sqrt{\sqrt{13}-1}-541755214884 \Big) \theta_1^2 \\
& + \left( 277870869900\sqrt{78(\sqrt{13}-1)}+13292409240\sqrt{78}-1412412683760\sqrt{6(\sqrt{13}-1)}+523921239060\sqrt{26(\sqrt{13}-1)} \right. \\
& -847778258520\sqrt{39(\sqrt{13}-1)}+2634079392360\sqrt{6}+896641545780\sqrt{26}+839890237068\sqrt{39} \\
& -2444157602640\sqrt{2(\sqrt{13}-1)}+2696013976968\sqrt{3(\sqrt{13}-1)}-820722286728\sqrt{13(\sqrt{13}-1)}+4696456510980\sqrt{2} \\
& +908276800860\sqrt{3}+1672043266260\sqrt{13}+2719834387920\sqrt{\sqrt{13}-1}+3497002028676 \Big) \theta_1 \\
& -22644233664\sqrt{78(\sqrt{13}-1)}-660127107564\sqrt{78}+771939362064\sqrt{6(\sqrt{13}-1)}-1589590378956\sqrt{26(\sqrt{13}-1)} \\
& +568133682072\sqrt{39(\sqrt{13}-1)}-10415072496348\sqrt{6}-525173105628\sqrt{26}-501494309424\sqrt{39} \\
& +9100285213188\sqrt{2(\sqrt{13}-1)}-201954535536\sqrt{3(\sqrt{13}-1)}+2011648936104\sqrt{13(\sqrt{13}-1)} \\
& -24037501737132\sqrt{2}-9983913007536\sqrt{3}+1393986824424\sqrt{13}-8182103383584\sqrt{\sqrt{13}-1}-5205649741032
\end{aligned}$$

$$\begin{aligned}
v_6 = & \left( \left( -16454168751\sqrt{312(4-\sqrt{13})}-47390453990\sqrt{78(\sqrt{13}-1)}-150615625836\sqrt{78(\sqrt{13}-3)} \right. \right. \\
& -476309407899\sqrt{24(4-\sqrt{13})}-178740295623\sqrt{104(4-\sqrt{13})}+6550417786\sqrt{156(4-\sqrt{13})}-200574075210\sqrt{78} \\
& -2171251706\sqrt{6(\sqrt{13}-1)}-7529062602\sqrt{26(\sqrt{13}-1)}-57774655044\sqrt{39(\sqrt{13}-1)} \\
& -214365468304\sqrt{6(\sqrt{13}-3)}-296443549050\sqrt{26(\sqrt{13}-3)}-292212076266\sqrt{39(\sqrt{13}-3)} \\
& -1462364383779\sqrt{8(4-\sqrt{13})}+32306974726\sqrt{12(4-\sqrt{13})}-288116397432\sqrt{52(4-\sqrt{13})}-920784383298\sqrt{6} \\
& -65280647370\sqrt{26}-236802250056\sqrt{39}-18984505410\sqrt{2(\sqrt{13}-1)}+89743882176\sqrt{3(\sqrt{13}-1)} \\
& -64219197876\sqrt{13(\sqrt{13}-1)}-1210153009962\sqrt{2(\sqrt{13}-3)}-854131510830\sqrt{3(\sqrt{13}-3)} \\
& -564912731370\sqrt{13(\sqrt{13}-3)}-886418028756\sqrt{4(4-\sqrt{13})}-3175132080354\sqrt{2}-2214035946096\sqrt{3} \\
& -782921827236\sqrt{13}+331481255040\sqrt{\sqrt{13}-1}-2116117476798\sqrt{\sqrt{13}-3}-1175644168764 \Big) \theta_1^2 \\
& + \left( -261285520014\sqrt{312(4-\sqrt{13})}-217409051218\sqrt{78(\sqrt{13}-1)}-622316394570\sqrt{78(\sqrt{13}-3)} \right. \\
& -1400420673054\sqrt{24(4-\sqrt{13})}+77047066524\sqrt{104(4-\sqrt{13})}-633787226572\sqrt{156(4-\sqrt{13})} \\
& -611561015256\sqrt{78}+582751308854\sqrt{6(\sqrt{13}-1)}-11268257814\sqrt{26(\sqrt{13}-1)}-151941836640\sqrt{39(\sqrt{13}-1)} \\
& -1715193142130\sqrt{6(\sqrt{13}-3)}-455563942290\sqrt{26(\sqrt{13}-3)}-926400471348\sqrt{39(\sqrt{13}-3)} \\
& -1572951646968\sqrt{8(4-\sqrt{13})}-1182760486024\sqrt{12(4-\sqrt{13})}-491208218736\sqrt{52(4-\sqrt{13})} \\
& -2382877556280\sqrt{6}-525308225280\sqrt{26}-750058129428\sqrt{39}-127877818926\sqrt{2(\sqrt{13}-1)} \\
& +264092706384\sqrt{3(\sqrt{13}-1)}-100408016484\sqrt{13(\sqrt{13}-1)}-1885983791250\sqrt{2(\sqrt{13}-3)} \\
& -2558306038236\sqrt{3(\sqrt{13}-3)}-814865740884\sqrt{13(\sqrt{13}-3)}-152263837284\sqrt{4(4-\sqrt{13})}-5651397155352\sqrt{2} \\
& -2529045217452\sqrt{3}-2261705398908\sqrt{13}-1107438540\sqrt{\sqrt{13}-1}-2431312258596\sqrt{\sqrt{13}-3}
\end{aligned}$$

$$\begin{aligned}
 & -2394241466916) \Theta_1 + 76485755454\sqrt{312(4-\sqrt{13})} + 18112893910\sqrt{78(\sqrt{13}-1)} + 833223741870\sqrt{78(\sqrt{13}-3)} \\
 & + 4194444816042\sqrt{24(4-\sqrt{13})} + 2615948219322\sqrt{104(4-\sqrt{13})} + 612061589596\sqrt{156(4-\sqrt{13})} \\
 & + 1194861591888\sqrt{78+1790482702138\sqrt{6(\sqrt{13}-1)}} + 522547672758\sqrt{26(\sqrt{13}-1)} \\
 & + 856190279844\sqrt{39(\sqrt{13}-1)} + 1475794800974\sqrt{6(\sqrt{13}-3)} + 2058789844194\sqrt{26(\sqrt{13}-3)} \\
 & + 2449219436712\sqrt{39(\sqrt{13}-3)} + 12685094068842\sqrt{8(4-\sqrt{13})} + 1362249350176\sqrt{12(4-\sqrt{13})} \\
 & + 1458981200556\sqrt{52(4-\sqrt{13})} + 3249228240648\sqrt{6} - 115720418712\sqrt{26} + 127070102184\sqrt{39} \\
 & - 1349248210830\sqrt{2(\sqrt{13}-1)} - 2477933969940\sqrt{3(\sqrt{13}-1)} + 9103835640\sqrt{13(\sqrt{13}-1)} \\
 & + 6757060637730\sqrt{2(\sqrt{13}-3)} + 6316192474800\sqrt{3(\sqrt{13}-3)} + 4731647487060\sqrt{13(\sqrt{13}-3)} \\
 & + 8516766878568\sqrt{4(4-\sqrt{13})} + 13033338017544\sqrt{2} + 24092955961152\sqrt{3} + 3147389886912\sqrt{13} \\
 & + 1384757681112\sqrt{\sqrt{13}-1} + 21043552558236\sqrt{\sqrt{13}-3} + 3686540539032) i + (-131613425423\sqrt{312(4-\sqrt{13})} \\
 & + 88551276122\sqrt{78(\sqrt{13}-1)} - 142142839110\sqrt{78(\sqrt{13}-3)} - 375887075759\sqrt{24(4-\sqrt{13})} \\
 & - 453389075337\sqrt{104(4-\sqrt{13})} - 231627929636\sqrt{156(4-\sqrt{13})} + 21828322638\sqrt{78} - 481486852054\sqrt{6(\sqrt{13}-1)} \\
 & + 198169198230\sqrt{26(\sqrt{13}-1)} - 126014805704\sqrt{39(\sqrt{13}-1)} - 448454701006\sqrt{6(\sqrt{13}-3)} \\
 & - 581300086716\sqrt{26(\sqrt{13}-3)} + 18771974994\sqrt{39(\sqrt{13}-3)} - 1728489498177\sqrt{8(4-\sqrt{13})} \\
 & - 752422831472\sqrt{12(4-\sqrt{13})} - 615432666246\sqrt{52(4-\sqrt{13})} + 279497636574\sqrt{6} + 351661391322\sqrt{26} \\
 & + 212102566188\sqrt{39} - 1208292243366\sqrt{2(\sqrt{13}-1)} + 209569880260\sqrt{3(\sqrt{13}-1)} - 19112768400\sqrt{13(\sqrt{13}-1)} \\
 & - 1996399994160\sqrt{2(\sqrt{13}-3)} - 759696906722\sqrt{3(\sqrt{13}-3)} - 505441691178\sqrt{13(\sqrt{13}-3)} \\
 & - 2891717276586\sqrt{4(4-\sqrt{13})} + 411912780474\sqrt{2} - 63843827580\sqrt{3} + 3662516664\sqrt{13} \\
 & - 701727865644\sqrt{\sqrt{13}-1} - 3024753476550\sqrt{\sqrt{13}-3} + 234414217440) \Theta_1^2 + (-485953705408\sqrt{312(4-\sqrt{13})} \\
 & + 77314369138\sqrt{78(\sqrt{13}-1)} - 536153615922\sqrt{78(\sqrt{13}-3)} - 1762174555492\sqrt{24(4-\sqrt{13})} \\
 & - 865570530858\sqrt{104(4-\sqrt{13})} - 591990607756\sqrt{156(4-\sqrt{13})} - 76032122904\sqrt{78} - 975655405790\sqrt{6(\sqrt{13}-1)} \\
 & + 338796465690\sqrt{26(\sqrt{13}-1)} - 553054838164\sqrt{39(\sqrt{13}-1)} - 1142564272538\sqrt{6(\sqrt{13}-3)} \\
 & - 866657902062\sqrt{26(\sqrt{13}-3)} - 467801756460\sqrt{39(\sqrt{13}-3)} - 3797772147546\sqrt{8(4-\sqrt{13})} \\
 & - 2110836625768\sqrt{12(4-\sqrt{13})} - 870804592512\sqrt{52(4-\sqrt{13})} + 872674965552\sqrt{6} + 1088385539448\sqrt{26} \\
 & + 831899830620\sqrt{39} - 2039495559894\sqrt{2(\sqrt{13}-1)} + 1137607262324\sqrt{3(\sqrt{13}-1)} - 470657563224\sqrt{13(\sqrt{13}-1)} \\
 & - 1327088115054\sqrt{2(\sqrt{13}-3)} - 3498722425900\sqrt{3(\sqrt{13}-3)} - 1058047600620\sqrt{13(\sqrt{13}-3)} \\
 & - 4507866532572\sqrt{4(4-\sqrt{13})} - 719522191128\sqrt{2} - 304693140492\sqrt{3} - 160741716228\sqrt{13} \\
 & + 808216646952\sqrt{\sqrt{13}-1} - 4988505234180\sqrt{\sqrt{13}-3} + 2487569153940) \Theta_1 + 656255268622\sqrt{312(4-\sqrt{13})} \\
 & - 351457959010\sqrt{78(\sqrt{13}-1)} + 236345270010\sqrt{78(\sqrt{13}-3)} + 726791099086\sqrt{24(4-\sqrt{13})} \\
 & + 3337508108166\sqrt{104(4-\sqrt{13})} + 925162539748\sqrt{156(4-\sqrt{13})} - 470321980896\sqrt{78} \\
 & + 2029375456394\sqrt{6(\sqrt{13}-1)} - 1699183330206\sqrt{26(\sqrt{13}-1)} + 198562829152\sqrt{39(\sqrt{13}-1)} \\
 & + 2426964564362\sqrt{6(\sqrt{13}-3)} + 1152493002762\sqrt{26(\sqrt{13}-3)} - 780968811684\sqrt{39(\sqrt{13}-3)} \\
 & + 13567726519386\sqrt{8(4-\sqrt{13})} + 3503131757008\sqrt{12(4-\sqrt{13})} + 986748236148\sqrt{52(4-\sqrt{13})} \\
 & - 1333270036176\sqrt{6} - 3105402077496\sqrt{26} - 2125863620448\sqrt{39} + 9982957572510\sqrt{2(\sqrt{13}-1)} \\
 & + 36938218720\sqrt{3(\sqrt{13}-1)} - 210082066860\sqrt{13(\sqrt{13}-1)} + 2482683986250\sqrt{2(\sqrt{13}-3)} \\
 & + 796530472660\sqrt{3(\sqrt{13}-3)} + 3521848188000\sqrt{13(\sqrt{13}-3)} + 6897583443504\sqrt{4(4-\sqrt{13})} + 9315916177152\sqrt{2} \\
 & + 3878857592376\sqrt{3} - 862614219384\sqrt{13} + 3150102399756\sqrt{\sqrt{13}-1} + 23829222568008\sqrt{\sqrt{13}-3} \\
 & + 7780537270896 \\
 v_7 = & \left( (-253154309232\sqrt{78} + 394679942592\sqrt{6} + 505183239936\sqrt{26} - 262142036160\sqrt{39} - 2716832029392\sqrt{2} \right. \\
 & + 389879315904\sqrt{3} - 1359667929888\sqrt{13} + 2591889985152) \Theta_1^2 + (-1175219208528\sqrt{78} - 952012993776\sqrt{6} \\
 & + 1892714775696\sqrt{26} - 748097128992\sqrt{39} - 6672327878736\sqrt{2} + 863948252544\sqrt{3} - 3174373318464\sqrt{13} \\
 & + 9353532198816) \Theta_1 + 4151926351296\sqrt{78} - 13206008607168\sqrt{6} + 465755406912\sqrt{26} + 1383690571776\sqrt{39} \\
 & + 16405205451648\sqrt{2} - 2325104748096\sqrt{3} + 9972213917184\sqrt{13} - 24636924990912) i + (-248511766032\sqrt{78} \\
 & - 2779527628800\sqrt{6} - 947400777696\sqrt{26} - 666179550912\sqrt{39} - 4682519685168\sqrt{2} - 2688556007904\sqrt{3} \\
 & - 1641561050688\sqrt{13} - 11753751741888) \Theta_1^2 + (-1666747572912\sqrt{78} - 6184981185552\sqrt{6} \\
 & - 2262257806128\sqrt{26} - 2203108236384\sqrt{39} - 9889057920144\sqrt{2} - 7358001283776\sqrt{3} - 4269311177664\sqrt{13} \\
 & - 10630373608608) \Theta_1 + 643972212288\sqrt{78} + 16604141052096\sqrt{6} + 4233554874816\sqrt{26} + 2557102080768\sqrt{39} \\
 & + 23684529146304\sqrt{2} + 13190200637952\sqrt{3} + 6648778523712\sqrt{13} + 74430301830144
 \end{aligned}$$

$$\begin{aligned}
v_8 = & \left( (16454168751\sqrt{312(4-\sqrt{13})}-47390453990\sqrt{78(\sqrt{13}-1)}-150615625836\sqrt{78(\sqrt{13}-3)} \right. \\
& +476309407899\sqrt{24(4-\sqrt{13})}+178740295623\sqrt{104(4-\sqrt{13})}-6550417786\sqrt{156(4-\sqrt{13})}+200574075210\sqrt{78} \\
& -2171251706\sqrt{6(\sqrt{13}-1)}-7529062602\sqrt{26(\sqrt{13}-1)}-57774655044\sqrt{39(\sqrt{13}-1)} \\
& -214365468304\sqrt{6(\sqrt{13}-3)}-296443549050\sqrt{26(\sqrt{13}-3)}-292212076266\sqrt{39(\sqrt{13}-3)} \\
& +1462364383779\sqrt{8(4-\sqrt{13})}-32306974726\sqrt{12(4-\sqrt{13})}+288116397432\sqrt{52(4-\sqrt{13})}+920784383298\sqrt{6} \\
& +65280647370\sqrt{26}+236802250056\sqrt{39}-18984505410\sqrt{2(\sqrt{13}-1)}+89743882176\sqrt{3(\sqrt{13}-1)} \\
& -64219197876\sqrt{13(\sqrt{13}-1)}-1210153009962\sqrt{2(\sqrt{13}-3)}-854131510830\sqrt{3(\sqrt{13}-3)} \\
& -564912731370\sqrt{13(\sqrt{13}-3)}+886418028756\sqrt{4(4-\sqrt{13})}+3175132080354\sqrt{2}+2214035946096\sqrt{3} \\
& +782921827236\sqrt{13}+331481255040\sqrt{\sqrt{13}-1}-2116117476798\sqrt{\sqrt{13}-3}+1175644168764)\Theta_1^2 \\
& + (261285520014\sqrt{312(4-\sqrt{13})}-217409051218\sqrt{78(\sqrt{13}-1)}-622316394570\sqrt{78(\sqrt{13}-3)} \\
& +1400420673054\sqrt{24(4-\sqrt{13})}-77047066524\sqrt{104(4-\sqrt{13})}+633787226572\sqrt{156(4-\sqrt{13})} \\
& +611561015256\sqrt{78}+582751308854\sqrt{6(\sqrt{13}-1)}-11268257814\sqrt{26(\sqrt{13}-1)}-151941836640\sqrt{39(\sqrt{13}-1)} \\
& -1715193142130\sqrt{6(\sqrt{13}-3)}-455563942290\sqrt{26(\sqrt{13}-3)}-926400471348\sqrt{39(\sqrt{13}-3)} \\
& +1572951646968\sqrt{8(4-\sqrt{13})}+1182760486024\sqrt{12(4-\sqrt{13})}+491208218736\sqrt{52(4-\sqrt{13})} \\
& +2382877556280\sqrt{6}+525308225280\sqrt{26}+750058129428\sqrt{39}-127877818926\sqrt{2(\sqrt{13}-1)} \\
& +264092706384\sqrt{3(\sqrt{13}-1)}-100408016484\sqrt{13(\sqrt{13}-1)}-1885983791250\sqrt{2(\sqrt{13}-3)} \\
& -2558306038236\sqrt{3(\sqrt{13}-3)}-814865740884\sqrt{13(\sqrt{13}-3)}+152263837284\sqrt{4(4-\sqrt{13})}+5651397155352\sqrt{2} \\
& +2529045217452\sqrt{3}+2261705398908\sqrt{13}-1107438540\sqrt{\sqrt{13}-1}-2431312258596\sqrt{\sqrt{13}-3} \\
& +2394241466916)\Theta_1-76485755454\sqrt{312(4-\sqrt{13})}+18112893910\sqrt{78(\sqrt{13}-1)}+833223741870\sqrt{78(\sqrt{13}-3)} \\
& -4194444816042\sqrt{24(4-\sqrt{13})}-2615948219322\sqrt{104(4-\sqrt{13})}-612061589596\sqrt{156(4-\sqrt{13})} \\
& -1194861591888\sqrt{78}+1790482702138\sqrt{6(\sqrt{13}-1)}+522547672758\sqrt{26(\sqrt{13}-1)} \\
& +856190279844\sqrt{39(\sqrt{13}-1)}+1475794800974\sqrt{6(\sqrt{13}-3)}+2058789844194\sqrt{26(\sqrt{13}-3)} \\
& +2449219436712\sqrt{39(\sqrt{13}-3)}-12685094068842\sqrt{8(4-\sqrt{13})}-1362249350176\sqrt{12(4-\sqrt{13})} \\
& -1458981200556\sqrt{52(4-\sqrt{13})}-3249228240648\sqrt{6}+115720418712\sqrt{26}-127070102184\sqrt{39} \\
& -1349248210830\sqrt{2(\sqrt{13}-1)}-2477933969940\sqrt{3(\sqrt{13}-1)}+9103835640\sqrt{13(\sqrt{13}-1)} \\
& +6757060637730\sqrt{2(\sqrt{13}-3)}+6316192474800\sqrt{3(\sqrt{13}-3)}+4731647487060\sqrt{13(\sqrt{13}-3)} \\
& -8516766878568\sqrt{4(4-\sqrt{13})}-13033338017544\sqrt{2}-24092955961152\sqrt{3}-3147389886912\sqrt{13} \\
& +1384757681112\sqrt{\sqrt{13}-1}+21043552558236\sqrt{\sqrt{13}-3}-3686540539032)\Theta_1^2 \\
& + (131613425423\sqrt{312(4-\sqrt{13})}+88551276122\sqrt{78(\sqrt{13}-1)}-142142839110\sqrt{78(\sqrt{13}-3)} \\
& +375887075759\sqrt{24(4-\sqrt{13})}+453389075337\sqrt{104(4-\sqrt{13})}+231627929636\sqrt{156(4-\sqrt{13})} \\
& -21828322638\sqrt{78}-481486852054\sqrt{6(\sqrt{13}-1)}+198169198230\sqrt{26(\sqrt{13}-1)}-126014805704\sqrt{39(\sqrt{13}-1)} \\
& -448454701006\sqrt{6(\sqrt{13}-3)}-581300086716\sqrt{26(\sqrt{13}-3)}+18771974994\sqrt{39(\sqrt{13}-3)} \\
& +1728489498177\sqrt{8(4-\sqrt{13})}+752422831472\sqrt{12(4-\sqrt{13})}+615432666246\sqrt{52(4-\sqrt{13})}-279497636574\sqrt{6} \\
& -351661391322\sqrt{26}-212102566188\sqrt{39}-1208292243366\sqrt{2(\sqrt{13}-1)}+209569880260\sqrt{3(\sqrt{13}-1)} \\
& -19112768400\sqrt{13(\sqrt{13}-1)}-1996399994160\sqrt{2(\sqrt{13}-3)}-759696906722\sqrt{3(\sqrt{13}-3)} \\
& -505441691178\sqrt{13(\sqrt{13}-3)}+2891717726586\sqrt{4(4-\sqrt{13})}-411912780474\sqrt{2}+63843827580\sqrt{3} \\
& -3662516664\sqrt{13}-701727865644\sqrt{\sqrt{13}-1}-3024753476550\sqrt{\sqrt{13}-3}-234414217440)\Theta_1^2 \\
& + (485953705408\sqrt{312(4-\sqrt{13})}+77314369138\sqrt{78(\sqrt{13}-1)}-536153615922\sqrt{78(\sqrt{13}-3)} \\
& +1762174555492\sqrt{24(4-\sqrt{13})}+865570530858\sqrt{104(4-\sqrt{13})}+591990607756\sqrt{156(4-\sqrt{13})} \\
& +76032122904\sqrt{78}-975655405790\sqrt{6(\sqrt{13}-1)}+338796465690\sqrt{26(\sqrt{13}-1)}-553054838164\sqrt{39(\sqrt{13}-1)} \\
& -1142564272538\sqrt{6(\sqrt{13}-3)}-866657902062\sqrt{26(\sqrt{13}-3)}-467801756460\sqrt{39(\sqrt{13}-3)} \\
& +3797772147546\sqrt{8(4-\sqrt{13})}+2110836625768\sqrt{12(4-\sqrt{13})}+870804592512\sqrt{52(4-\sqrt{13})}-872674965552\sqrt{6} \\
& -1088385539448\sqrt{26}-831899830620\sqrt{39}-2039495559894\sqrt{2(\sqrt{13}-1)}+1137607262324\sqrt{3(\sqrt{13}-1)} \\
& -470657563224\sqrt{13(\sqrt{13}-1)}-1327088115054\sqrt{2(\sqrt{13}-3)}-3498722425900\sqrt{3(\sqrt{13}-3)} \\
& -1058047600620\sqrt{13(\sqrt{13}-3)}+4507866532572\sqrt{4(4-\sqrt{13})}+719522191128\sqrt{2}+304693140492\sqrt{3} \\
& +160741716228\sqrt{13}+808216646952\sqrt{\sqrt{13}-1}-4988505234180\sqrt{\sqrt{13}-3}-2487569153940)\Theta_1 \\
& -656255268622\sqrt{312(4-\sqrt{13})}-351457959010\sqrt{78(\sqrt{13}-1)}+236345270010\sqrt{78(\sqrt{13}-3)} \\
& -726791099086\sqrt{24(4-\sqrt{13})}-3337508108166\sqrt{104(4-\sqrt{13})}-925162539748\sqrt{156(4-\sqrt{13})} \\
& +470321980896\sqrt{78}+2029375456394\sqrt{6(\sqrt{13}-1)}-1699183330206\sqrt{26(\sqrt{13}-1)}
\end{aligned}$$

$$\begin{aligned}
 &+198562829152\sqrt{39(\sqrt{13}-1)}+2426964564362\sqrt{6(\sqrt{13}-3)}+1152493002762\sqrt{26(\sqrt{13}-3)} \\
 &-780968811684\sqrt{39(\sqrt{13}-3)}-13567726519386\sqrt{8(4-\sqrt{13})}-3503131757008\sqrt{12(4-\sqrt{13})} \\
 &-986748236148\sqrt{52(4-\sqrt{13})}+1333270036176\sqrt{6}+3105402077496\sqrt{26}+2125863620448\sqrt{39} \\
 &+9982957572510\sqrt{2(\sqrt{13}-1)}+36938218720\sqrt{3(\sqrt{13}-1)}-210082066860\sqrt{13(\sqrt{13}-1)} \\
 &+2482683986250\sqrt{2(\sqrt{13}-3)}+796530472660\sqrt{3(\sqrt{13}-3)}+3521848188000\sqrt{13(\sqrt{13}-3)} \\
 &-6897583443504\sqrt{4(4-\sqrt{13})}-9315916177152\sqrt{2}-3878857592376\sqrt{3}+862614219384\sqrt{13} \\
 &+3150102399756\sqrt{\sqrt{13}-1}+23829222568008\sqrt{\sqrt{13}-3}-7780537270896
 \end{aligned}$$

$$\begin{aligned}
 v_9 = &\left( \left( -44504276384\sqrt{78(\sqrt{13}-1)}+318237636924\sqrt{78}+151828132924\sqrt{6(\sqrt{13}-1)} \right. \right. \\
 &-34136920536\sqrt{26(\sqrt{13}-1)}+98935477176\sqrt{39(\sqrt{13}-1)}+242322009228\sqrt{6}+931345797708\sqrt{26} \\
 &-186392234436\sqrt{39}-166138799892\sqrt{2(\sqrt{13}-1)}-573401985936\sqrt{3(\sqrt{13}-1)}+254859333504\sqrt{13(\sqrt{13}-1)} \\
 &-58488656148\sqrt{2}+1175430629724\sqrt{3}-215870718492\sqrt{13}-1558758003816\sqrt{\sqrt{13}-1}+3336815059812 \Big) \theta_1^2 \\
 &+ \left( -135089286184\sqrt{78(\sqrt{13}-1)}+691888522860\sqrt{78}+118098675500\sqrt{6(\sqrt{13}-1)} \right. \\
 &-274264532040\sqrt{26(\sqrt{13}-1)}+11847154560\sqrt{39(\sqrt{13}-1)}-1464891106068\sqrt{6}+2314315973448\sqrt{26} \\
 &-175448592660\sqrt{39}+531432423132\sqrt{2(\sqrt{13}-1)}-656996803320\sqrt{3(\sqrt{13}-1)}+427936224360\sqrt{13(\sqrt{13}-1)} \\
 &-3688428786120\sqrt{2}+5409811905420\sqrt{3}+69175002780\sqrt{13}-2166265940280\sqrt{\sqrt{13}-1}+10349226330300 \Big) \theta_1 \\
 &+509263660588\sqrt{78(\sqrt{13}-1)}-938070648468\sqrt{78}-3010980577748\sqrt{6(\sqrt{13}-1)}-623036788368\sqrt{26(\sqrt{13}-1)} \\
 &-1189535344944\sqrt{39(\sqrt{13}-1)}+7379540374284\sqrt{6}-3958070812236\sqrt{26}-1152333318408\sqrt{39} \\
 &+3616678251264\sqrt{2(\sqrt{13}-1)}+6297792128472\sqrt{3(\sqrt{13}-1)}-1185739493088\sqrt{13(\sqrt{13}-1)} \\
 &-9563947783884\sqrt{2}+5672324504808\sqrt{3}-1976154581232\sqrt{13}+7248951680568\sqrt{\sqrt{13}-1}-10695965318064 \Big) i \\
 &+ \left( -122671351452\sqrt{78(\sqrt{13}-1)}+142230668148\sqrt{78}+541399851096\sqrt{6(\sqrt{13}-1)} \right. \\
 &-175615983492\sqrt{26(\sqrt{13}-1)}+261956535816\sqrt{39(\sqrt{13}-1)}+2240309341764\sqrt{6}+118086900084\sqrt{26} \\
 &+336823402572\sqrt{39}+691687683024\sqrt{2(\sqrt{13}-1)}-688885480608\sqrt{3(\sqrt{13}-1)}+224811029232\sqrt{13(\sqrt{13}-1)} \\
 &+2371273089588\sqrt{2}+1274473966428\sqrt{3}+212079746988\sqrt{13}-487579872312\sqrt{\sqrt{13}-1}-541755214884 \Big) \theta_1^2 \\
 &+ \left( -277870869900\sqrt{78(\sqrt{13}-1)}+13292409240\sqrt{78}+1412412683760\sqrt{6(\sqrt{13}-1)} \right. \\
 &-523921239060\sqrt{26(\sqrt{13}-1)}+847778258520\sqrt{39(\sqrt{13}-1)}+2634079392360\sqrt{6}+896641545780\sqrt{26} \\
 &+839890237068\sqrt{39}+2444157602640\sqrt{2(\sqrt{13}-1)}-2696013976968\sqrt{3(\sqrt{13}-1)}+820722286728\sqrt{13(\sqrt{13}-1)} \\
 &+4696456510980\sqrt{2}+908276800860\sqrt{3}+1672043266260\sqrt{13}-2719834387920\sqrt{\sqrt{13}-1}+3497002028676 \Big) \theta_1 \\
 &+22644233664\sqrt{78(\sqrt{13}-1)}-660127107564\sqrt{78}-771939362064\sqrt{6(\sqrt{13}-1)}+1589590378956\sqrt{26(\sqrt{13}-1)} \\
 &-568133682072\sqrt{39(\sqrt{13}-1)}-10415072496348\sqrt{6}-525173105628\sqrt{26}-501494309424\sqrt{39} \\
 &-9100285213188\sqrt{2(\sqrt{13}-1)}+201954535536\sqrt{3(\sqrt{13}-1)}-2011648936104\sqrt{13(\sqrt{13}-1)} \\
 &-24037501737132\sqrt{2}-9983913007536\sqrt{3}+1393986824424\sqrt{13}+8182103383584\sqrt{\sqrt{13}-1}-5205649741032
 \end{aligned}$$

$$\begin{aligned}
 v_{10} = &\left( \left( -731915712552\sqrt{78(\sqrt{13}-3)}-125519540184\sqrt{78}-2210765462464\sqrt{6(\sqrt{13}-3)} \right. \right. \\
 &-722872066380\sqrt{26(\sqrt{13}-3)}-797653767444\sqrt{39(\sqrt{13}-3)}-458077199424\sqrt{6}-323269220712\sqrt{26} \\
 &+366532023072\sqrt{39}-2555517112980\sqrt{2(\sqrt{13}-3)}-3878884987380\sqrt{3(\sqrt{13}-3)}-508596806388\sqrt{13(\sqrt{13}-3)} \\
 &+108644753568\sqrt{2}+197578285008\sqrt{3}+100156084800\sqrt{13}-4395208196964\sqrt{\sqrt{13}-3}-911613168960 \Big) \theta_1^2 \\
 &+ \left( -1488974296632\sqrt{78(\sqrt{13}-3)}-252967696776\sqrt{78}-3042281257184\sqrt{6(\sqrt{13}-3)} \right. \\
 &-2064024790056\sqrt{26(\sqrt{13}-3)}-1984448071968\sqrt{39(\sqrt{13}-3)}+1522918860216\sqrt{6}-466687258656\sqrt{26} \\
 &+1039084227960\sqrt{39}-5313676608864\sqrt{2(\sqrt{13}-3)}-7546811272416\sqrt{3(\sqrt{13}-3)} \\
 &-22182711010264\sqrt{13(\sqrt{13}-3)}+1373559746208\sqrt{2}-3006982158648\sqrt{3}-724293724632\sqrt{13} \\
 &-12927479536296\sqrt{\sqrt{13}-3}-2976810231720 \Big) \theta_1 + 1985716744632\sqrt{78(\sqrt{13}-3)}+127930187208\sqrt{78} \\
 &+3958478787224\sqrt{6(\sqrt{13}-3)}+2767825654224\sqrt{26(\sqrt{13}-3)}+5971067624712\sqrt{39(\sqrt{13}-3)} \\
 &-2083313767704\sqrt{6}+462949809576\sqrt{26}-1710278238192\sqrt{39}+14037954330816\sqrt{2(\sqrt{13}-3)} \\
 &+30145415042808\sqrt{3(\sqrt{13}-3)}+2388741052008\sqrt{13(\sqrt{13}-3)}-7723861751736\sqrt{2}-15805866028368\sqrt{3} \\
 &+127993749792\sqrt{13}+23433143976216\sqrt{\sqrt{13}-3}+635011026624 \Big) i + (154300709940\sqrt{78(\sqrt{13}-3)} \\
 &+269227309776\sqrt{78}+761698308956\sqrt{6(\sqrt{13}-3)}-129453209208\sqrt{26(\sqrt{13}-3)}+583684706364\sqrt{39(\sqrt{13}-3)} \\
 &-1490196729528\sqrt{6}+270624017040\sqrt{26}-411101580336\sqrt{39}-1353303589248\sqrt{2(\sqrt{13}-3)}
 \end{aligned}$$

$$\begin{aligned}
 &+1356420570076\sqrt{3(\sqrt{13}-3)}+371194537620\sqrt{13(\sqrt{13}-3)}+1457459351400\sqrt{2}+1424901505872\sqrt{3} \\
 &+327820027392\sqrt{13}-462358944060\sqrt{\sqrt{13}-3}+2700029672496) \Theta_1^2+(-80589673632\sqrt{78(\sqrt{13}-3)} \\
 &+1121899025424\sqrt{78}+743419518712\sqrt{6(\sqrt{13}-3)}+1000291281648\sqrt{26(\sqrt{13}-3)} \\
 &+347063984424\sqrt{39(\sqrt{13}-3)}-1808068270320\sqrt{6}+68981441880\sqrt{26}-571067720616\sqrt{39} \\
 &+3818491311336\sqrt{2(\sqrt{13}-3)}-1067410167304\sqrt{3(\sqrt{13}-3)}+1721153813424\sqrt{13(\sqrt{13}-3)} \\
 &-1165371165192\sqrt{2}+5157949405128\sqrt{3}+229994431992\sqrt{13}+2686412880528\sqrt{\sqrt{13}-3}+3521174897448) \Theta_1 \\
 &-1822444574184\sqrt{78(\sqrt{13}-3)}-952582707096\sqrt{78}-4330096073368\sqrt{6(\sqrt{13}-3)} \\
 &-1347178222848\sqrt{26(\sqrt{13}-3)}-5512616298744\sqrt{39(\sqrt{13}-3)}+3348263555208\sqrt{6}+417961305528\sqrt{26} \\
 &+1596753185184\sqrt{39}-1944700416672\sqrt{2(\sqrt{13}-3)}-20247022085576\sqrt{3(\sqrt{13}-3)} \\
 &-3825810122136\sqrt{13(\sqrt{13}-3)}-8604302416008\sqrt{2}-8486483594784\sqrt{3}-4430219876784\sqrt{13} \\
 &+4880645143608\sqrt{\sqrt{13}-3}-23270763245808
 \end{aligned}$$

$$\begin{aligned}
 v_{11} = &\left( (-139285184364\sqrt{78(\sqrt{13}-1)}+131930836488\sqrt{78}+147485629512\sqrt{6(\sqrt{13}-1)} \right. \\
 &-49195045740\sqrt{26(\sqrt{13}-1)}-16613832912\sqrt{39(\sqrt{13}-1)}+976183494312\sqrt{6}-295192621944\sqrt{26} \\
 &-68424768396\sqrt{39}-204107810712\sqrt{2(\sqrt{13}-1)}-393914221584\sqrt{3(\sqrt{13}-1)}+126420937752\sqrt{13(\sqrt{13}-1)} \\
 &-981357553800\sqrt{2}-316722878940\sqrt{3}+128818185732\sqrt{13}-895795493736\sqrt{\sqrt{13}-1}-1127988040140) \Theta_1^2 \\
 &+(-569907388620\sqrt{78(\sqrt{13}-1)}+309663347508\sqrt{78}+1283601293208\sqrt{6(\sqrt{13}-1)} \\
 &-296801047668\sqrt{26(\sqrt{13}-1)}-292036518720\sqrt{39(\sqrt{13}-1)}-926166269964\sqrt{6}-877793935176\sqrt{26} \\
 &-894131968788\sqrt{39}+275676785280\sqrt{2(\sqrt{13}-1)}-128811390552\sqrt{3(\sqrt{13}-1)}+227120191392\sqrt{13(\sqrt{13}-1)} \\
 &+2252150700696\sqrt{2}+429274655004\sqrt{3}+1747850662476\sqrt{13}-2168480817360\sqrt{\sqrt{13}-1}-2823852974436) \Theta_1 \\
 &+545489448408\sqrt{78(\sqrt{13}-1)}-1120807155228\sqrt{78}+569984826528\sqrt{6(\sqrt{13}-1)}+422058557148\sqrt{26(\sqrt{13}-1)} \\
 &+522845214744\sqrt{39(\sqrt{13}-1)}-1953716627148\sqrt{6}+1601894333772\sqrt{26}+810795451416\sqrt{39} \\
 &+918181829604\sqrt{2(\sqrt{13}-1)}+1341924188592\sqrt{3(\sqrt{13}-1)}-1167531821808\sqrt{13(\sqrt{13}-1)} \\
 &+17969533018476\sqrt{2}+7563645288888\sqrt{3}-806813224272\sqrt{13}+10018467042792\sqrt{\sqrt{13}-1}+9332185881408) i \\
 &+ (54431200792\sqrt{78(\sqrt{13}-1)}-260181139440\sqrt{78}-421573853012\sqrt{6(\sqrt{13}-1)}+220722412968\sqrt{26(\sqrt{13}-1)} \\
 &+9926924408\sqrt{39(\sqrt{13}-1)}+1330427190144\sqrt{6}+137956773480\sqrt{26}+562840700988\sqrt{39} \\
 &-1724896803708\sqrt{2(\sqrt{13}-1)}-269745720088\sqrt{3(\sqrt{13}-1)}+186585492432\sqrt{13(\sqrt{13}-1)}-1529086474968\sqrt{2} \\
 &-1611741636804\sqrt{3}-432827297340\sqrt{13}-1891035603600\sqrt{\sqrt{13}-1}+2072219149572) \Theta_1^2 \\
 &+(-123242131624\sqrt{78(\sqrt{13}-1)}-629423531592\sqrt{78}-538898127820\sqrt{6(\sqrt{13}-1)} \\
 &+153671692320\sqrt{26(\sqrt{13}-1)}-258331417808\sqrt{39(\sqrt{13}-1)}+2483733732792\sqrt{6}+510466739628\sqrt{26} \\
 &+1480412324364\sqrt{39}-1634833517148\sqrt{2(\sqrt{13}-1)}-420799452320\sqrt{3(\sqrt{13}-1)}-120592839720\sqrt{13(\sqrt{13}-1)} \\
 &-288249093444\sqrt{2}-3477795150804\sqrt{3}-1441005667356\sqrt{13}-1103401094016\sqrt{\sqrt{13}-1}+1046895667428) \Theta_1 \\
 &-680271684356\sqrt{78(\sqrt{13}-1)}+1535540157276\sqrt{78}+3286811550724\sqrt{6(\sqrt{13}-1)} \\
 &-1808776281456\sqrt{26(\sqrt{13}-1)}-171008023768\sqrt{39(\sqrt{13}-1)}+952537513356\sqrt{6}-2410441570884\sqrt{26} \\
 &-3236183383632\sqrt{39}+10865629931832\sqrt{2(\sqrt{13}-1)}+275830972976\sqrt{3(\sqrt{13}-1)} \\
 &-2431813069824\sqrt{13(\sqrt{13}-1)}+28226661439548\sqrt{2}+15801032567424\sqrt{3}+4492371086184\sqrt{13} \\
 &+14482308183096\sqrt{\sqrt{13}-1}-3903632907096
 \end{aligned}$$

$$\begin{aligned}
 v_{12} = &\left( (-16454168751\sqrt{312(4-\sqrt{13})}+47390453990\sqrt{78(\sqrt{13}-1)}-150615625836\sqrt{78(\sqrt{13}-3)} \right. \\
 &-476309407899\sqrt{24(4-\sqrt{13})}-178740295623\sqrt{104(4-\sqrt{13})}+6550417786\sqrt{156(4-\sqrt{13})}+200574075210\sqrt{78} \\
 &+2171251706\sqrt{6(\sqrt{13}-1)}+7529062602\sqrt{26(\sqrt{13}-1)}+57774655044\sqrt{39(\sqrt{13}-1)} \\
 &-214365468304\sqrt{6(\sqrt{13}-3)}-296443549050\sqrt{26(\sqrt{13}-3)}-292212076266\sqrt{39(\sqrt{13}-3)} \\
 &-1462364383779\sqrt{8(4-\sqrt{13})}+32306974726\sqrt{12(4-\sqrt{13})}-288116397432\sqrt{52(4-\sqrt{13})}+920784383298\sqrt{6} \\
 &+65280647370\sqrt{26}+236802250056\sqrt{39}+18984505410\sqrt{2(\sqrt{13}-1)}-89743882176\sqrt{3(\sqrt{13}-1)} \\
 &+64219197876\sqrt{13(\sqrt{13}-1)}-1210153009962\sqrt{2(\sqrt{13}-3)}-854131510830\sqrt{3(\sqrt{13}-3)} \\
 &-564912731370\sqrt{13(\sqrt{13}-3)}-886418028756\sqrt{4(4-\sqrt{13})}+3175132080354\sqrt{2}+2214035946096\sqrt{3} \\
 &+782921827236\sqrt{13}-331481255040\sqrt{\sqrt{13}-1}-2116117476798\sqrt{\sqrt{13}-3}+1175644168764) \Theta_1^2 \\
 &+(-261285520014\sqrt{312(4-\sqrt{13})}+217409051218\sqrt{78(\sqrt{13}-1)}-622316394570\sqrt{78(\sqrt{13}-3)}
 \end{aligned}$$

$$\begin{aligned}
& -1400420673054\sqrt{24(4-\sqrt{13})}+77047066524\sqrt{104(4-\sqrt{13})}-633787226572\sqrt{156(4-\sqrt{13})} \\
& +611561015256\sqrt{78-582751308854\sqrt{6(\sqrt{13}-1)}+11268257814\sqrt{26(\sqrt{13}-1)}+151941836640\sqrt{39(\sqrt{13}-1)}} \\
& -1715193142130\sqrt{6(\sqrt{13}-3)}-455563942290\sqrt{26(\sqrt{13}-3)}-926400471348\sqrt{39(\sqrt{13}-3)} \\
& -1572951646968\sqrt{8(4-\sqrt{13})}-1182760486024\sqrt{12(4-\sqrt{13})}-491208218736\sqrt{52(4-\sqrt{13})} \\
& +2382877556280\sqrt{6}+525308225280\sqrt{26}+750058129428\sqrt{39}+127877818926\sqrt{2(\sqrt{13}-1)} \\
& -264092706384\sqrt{3(\sqrt{13}-1)}+100408016484\sqrt{13(\sqrt{13}-1)}-1885983791250\sqrt{2(\sqrt{13}-3)} \\
& -2558306038236\sqrt{3(\sqrt{13}-3)}-814865740884\sqrt{13(\sqrt{13}-3)}-152263837284\sqrt{4(4-\sqrt{13})}+5651397155352\sqrt{2} \\
& +2529045217452\sqrt{3}+2261705398908\sqrt{13}+1107438540\sqrt{\sqrt{13}-1}-2431312258596\sqrt{\sqrt{13}-3} \\
& +2394241466916\left)\Theta_1+76485755454\sqrt{312(4-\sqrt{13})}-18112893910\sqrt{78(\sqrt{13}-1)}+833223741870\sqrt{78(\sqrt{13}-3)}\right. \\
& +4194444816042\sqrt{24(4-\sqrt{13})}+2615948219322\sqrt{104(4-\sqrt{13})}+612061589596\sqrt{156(4-\sqrt{13})} \\
& -1194861591888\sqrt{78-1790482702138\sqrt{6(\sqrt{13}-1)}-522547672758\sqrt{26(\sqrt{13}-1)}} \\
& -856190279844\sqrt{39(\sqrt{13}-1)}+1475794800974\sqrt{6(\sqrt{13}-3)}+2058789844194\sqrt{26(\sqrt{13}-3)} \\
& +2449219436712\sqrt{39(\sqrt{13}-3)}+12685094068842\sqrt{8(4-\sqrt{13})}+1362249350176\sqrt{12(4-\sqrt{13})} \\
& +1458981200556\sqrt{52(4-\sqrt{13})}-3249228240648\sqrt{6}+115720418712\sqrt{26}-127070102184\sqrt{39} \\
& +1349248210830\sqrt{2(\sqrt{13}-1)}+2477933969940\sqrt{3(\sqrt{13}-1)}-9103835640\sqrt{13(\sqrt{13}-1)} \\
& +6757060637730\sqrt{2(\sqrt{13}-3)}+6316192474800\sqrt{3(\sqrt{13}-3)}+4731647487060\sqrt{13(\sqrt{13}-3)} \\
& +8516766878568\sqrt{4(4-\sqrt{13})}-13033338017544\sqrt{2}-24092955961152\sqrt{3}-3147389886912\sqrt{13} \\
& -1384757681112\sqrt{\sqrt{13}-1}+21043552558236\sqrt{\sqrt{13}-3}-3686540539032\left)\right. i+\left.(-131613425423\sqrt{312(4-\sqrt{13})}\right. \\
& -88551276122\sqrt{78(\sqrt{13}-1)}-142142839110\sqrt{78(\sqrt{13}-3)}-375887075759\sqrt{24(4-\sqrt{13})} \\
& -453389075337\sqrt{104(4-\sqrt{13})}-231627929636\sqrt{156(4-\sqrt{13})}-21828322638\sqrt{78}+481486852054\sqrt{6(\sqrt{13}-1)} \\
& -198169198230\sqrt{26(\sqrt{13}-1)}+126014805704\sqrt{39(\sqrt{13}-1)}-448454701006\sqrt{6(\sqrt{13}-3)} \\
& -581300086716\sqrt{26(\sqrt{13}-3)}+18771974994\sqrt{39(\sqrt{13}-3)}-1728489498177\sqrt{8(4-\sqrt{13})} \\
& -752422831472\sqrt{12(4-\sqrt{13})}-615432666246\sqrt{52(4-\sqrt{13})}-279497636574\sqrt{6}-351661391322\sqrt{26} \\
& -212102566188\sqrt{39}+1208292243366\sqrt{2(\sqrt{13}-1)}-209569880260\sqrt{3(\sqrt{13}-1)}+19112768400\sqrt{13(\sqrt{13}-1)} \\
& -1996399994160\sqrt{2(\sqrt{13}-3)}-759696906722\sqrt{3(\sqrt{13}-3)}-505441691178\sqrt{13(\sqrt{13}-3)} \\
& -2891717726586\sqrt{4(4-\sqrt{13})}-411912780474\sqrt{2}+63843827580\sqrt{3}-3662516664\sqrt{13} \\
& +701727865644\sqrt{\sqrt{13}-1}-3024753476550\sqrt{\sqrt{13}-3}-234414217440\left)\Theta_1^2+\left(-485953705408\sqrt{312(4-\sqrt{13})}\right. \right. \\
& -77314369138\sqrt{78(\sqrt{13}-1)}-536153615922\sqrt{78(\sqrt{13}-3)}-1762174555492\sqrt{24(4-\sqrt{13})} \\
& -865570530858\sqrt{104(4-\sqrt{13})}-591990607756\sqrt{156(4-\sqrt{13})}+76032122904\sqrt{78}+975655405790\sqrt{6(\sqrt{13}-1)} \\
& -338796465690\sqrt{26(\sqrt{13}-1)}+553054838164\sqrt{39(\sqrt{13}-1)}-114256427538\sqrt{6(\sqrt{13}-3)} \\
& -866657902062\sqrt{26(\sqrt{13}-3)}-467801756460\sqrt{39(\sqrt{13}-3)}-3797772147546\sqrt{8(4-\sqrt{13})} \\
& -2110836625768\sqrt{12(4-\sqrt{13})}-870804592512\sqrt{52(4-\sqrt{13})}-872674965552\sqrt{6}-1088385539448\sqrt{26} \\
& -831899830620\sqrt{39}+2039495559894\sqrt{2(\sqrt{13}-1)}-1137607262324\sqrt{3(\sqrt{13}-1)}+470657563224\sqrt{13(\sqrt{13}-1)} \\
& -1327088115054\sqrt{2(\sqrt{13}-3)}-3498722425900\sqrt{3(\sqrt{13}-3)}-1058047600620\sqrt{13(\sqrt{13}-3)} \\
& -4507866532572\sqrt{4(4-\sqrt{13})}+719522191128\sqrt{2}+304693140492\sqrt{3}+160741716228\sqrt{13} \\
& -808216646952\sqrt{\sqrt{13}-1}-4988505234180\sqrt{\sqrt{13}-3}-2487569153940\left)\Theta_1+656255268622\sqrt{312(4-\sqrt{13})}\right. \\
& +351457959010\sqrt{78(\sqrt{13}-1)}+236345270010\sqrt{78(\sqrt{13}-3)}+726791099086\sqrt{24(4-\sqrt{13})} \\
& +3337508108166\sqrt{104(4-\sqrt{13})}+925162539748\sqrt{156(4-\sqrt{13})}+470321980896\sqrt{78} \\
& -2029375456394\sqrt{6(\sqrt{13}-1)}+1699183330206\sqrt{26(\sqrt{13}-1)}-198562829152\sqrt{39(\sqrt{13}-1)} \\
& +2426964564362\sqrt{6(\sqrt{13}-3)}+1152493002762\sqrt{26(\sqrt{13}-3)}-780968811684\sqrt{39(\sqrt{13}-3)} \\
& +13567726519386\sqrt{8(4-\sqrt{13})}+3503131757008\sqrt{12(4-\sqrt{13})}+986748236148\sqrt{52(4-\sqrt{13})} \\
& +1333270036176\sqrt{6}+3105402077496\sqrt{26}+2125863620448\sqrt{39}-9982957572510\sqrt{2(\sqrt{13}-1)} \\
& -36938218720\sqrt{3(\sqrt{13}-1)}+210082066860\sqrt{13(\sqrt{13}-1)}+2482683986250\sqrt{2(\sqrt{13}-3)} \\
& +796530472660\sqrt{3(\sqrt{13}-3)}+3521848188000\sqrt{13(\sqrt{13}-3)}+6897583443504\sqrt{4(4-\sqrt{13})}-9315916177152\sqrt{2} \\
& -3878857592376\sqrt{3}+862614219384\sqrt{13}-3150102399756\sqrt{\sqrt{13}-1}+23829222568008\sqrt{\sqrt{13}-3} \\
& -7780537270896
\end{aligned}$$

# Complexity of Comparing Monomials and Two Improvements of the Buchberger-Möller Algorithm

Samuel Lundqvist

Department of Mathematics  
Stockholm University  
SE-106 91 Stockholm  
Sweden

**Abstract.** We give a new algorithm for merging sorted lists of monomials. Together with a projection technique we obtain a new complexity bound for the Buchberger-Möller algorithm.

## 1 Introduction

Vanishing ideals of points are of interest in many fields of mathematics — they are used in coding theory, in interpolation problems and even in statistics. Recently, the vanishing ideal of a set of affine points has been studied in molecular biology [9].

The Buchberger-Möller algorithm [3], was proposed as a tool to make computations over vanishing ideals of points. When doing complexity studies of this algorithm, one has to deal with arithmetic operations over the ground field and monomial manipulations. The number of arithmetic operations is reported [4,10] to be proportional to  $nm^3$ , where  $n$  denotes the number of variables and  $m$  the number of points. The number of integer comparisons needed for the monomial manipulations is reported [4,10] to be proportional to  $n^2m^2$ . In the biological applications, the coefficients of the points takes values in a finite field  $\mathbb{Z}_p$  and one usually has  $m \ll n$ . Accordingly, one has begun searching for algorithms which are optimized for these situations. In [6], an algorithm which uses  $O(nm^2 + m^6)$  operations (arithmetic and integer operations are treated the same) is given, while in [7], the same authors sharpened this bound so that it reads  $O(nm^2 + pm^4 + pm^3 \log(pm))$  operations, where again, arithmetic and integer operations are treated the same.

In this paper, we first make a thorough study of the complexity of comparing monomials. We restrict our analysis to the admissible monomial orders on  $n$  indeterminates given by invertible matrices with  $\mathbb{Z}$ -coefficients. These orders associate to each monomial an  $n$ -vector of integers with the property that comparing two monomials is the same as comparing the  $n$ -vectors lexicographically. Although this is a restriction on the set of admissible monomial orders, we remark that earlier complexity studies [4,7,10] have been performed only on a much smaller set of admissible orders, e.g. lex, deglex and degrevlex.

To give bounds for the monomial manipulations, we study algorithms for comparing lexicographically sorted  $n$ -vectors and give a fast algorithm for merging lexicographically sorted lists of such vectors. Summation of ordered polynomials is one example of a situation where merge algorithms are used and thus, our merge algorithm could speed up the computation of  $S$ -polynomials during the computation of a Grbner basis with respect to an ideal given by generators.

We then focus on the Buchberger-Möller algorithm. We notice that the upper bound for the number of arithmetic operations during the algorithm as given in [4] can be sharpened to read

$$O(nm^2 + \min(m, n)m^3).$$

After introducing a projection technique we show that the upper bound for the time complexity of the monomial manipulation part can be lowered to

$$O(\min(m, n)m^2 \log(m))$$

using one of the monomial orders lex, deglex or degrevlex. Thus, our method has better complexity than both the standard Buchberger-Möller algorithm and the methods optimized for the situations where  $m \ll n$ . (Recall that the complexity of the original algorithm was  $O(n^2m^2)$  and measured in *integer comparisons*.) Our method is also more general than the methods in [6,7] since we do not assume that  $\mathbb{k}$  is a finite field. From the two bounds above, it follows that the bottleneck of the Buchberger-Möller algorithm is the arithmetic operations.

Finally, we use our methods to give new complexity results for the FGLM-algorithm [4] and for the algorithms concerning ideals defined by functionals given in [10].

Throughout the rest of the paper, we let  $S = \mathbb{k}[x_1, \dots, x_n]$  denote the polynomial ring in  $n$  variables over a field  $\mathbb{k}$ . The notion  $x^\alpha$ , where  $\alpha = (\alpha_1, \dots, \alpha_n)$ , will be used as short for  $x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ .

## 2 Monomial Manipulations

The main result in this section is that it is possible to merge two lexicographically sorted lists  $a$  and  $b$  of  $n$ -tuples in some set  $\Sigma$  with  $s$  and  $t$  elements respectively using at most  $\min(s, t) + n$  comparisons in  $\Sigma$  plus  $\max(s, t)$  comparisons in the set  $\{1, \dots, n\}$ . This is better than the expected  $\max(s, t)n$   $\Sigma$ -comparisons. We show how one can apply this result to merge sorted lists of monomials.

### 2.1 Complexity Model

In computer algebra, it is often implicitly assumed that address and index arithmetics can be performed in constant time. This means that reading a byte from any position in the memory is done in constant time, and reading  $k$  bytes is an  $O(k)$ -operation. Such a model has advantages over a Turing machine, since

it is easier to work with and even more realistic in the cases when the memory on a modern computer is enough to handle the input data. We will use this model. Let  $a$  and  $b$  be integers. With this model, summation of  $a$  and  $b$  is  $O(\log(\max(|a|, |b|)))$  and comparison of  $a$  and  $b$  is  $O(\log(\min(|a|, |b|)))$ . We will assume that multiplication of  $a$  and  $b$  is  $O(\log(|a|) \log(|b|))$ , although there are better bounds [5].

Inspired by [10], we will split the complexity studies in two parts. We will give arithmetic complexity for the arithmetic operations and time complexity for the monomial manipulations. We remark that the time complexity of performing  $f$  arithmetic operations always is at least  $O(f)$ . We do not deal with growth of coefficients in the arithmetic operations, but refer the reader to [4]. In [1], techniques are discussed when  $\mathbb{k} = \mathbb{Q}$  using the Chinese remainder theorem.

### 2.2 Monomial Orders

An admissible monomial order  $\prec$  on  $x_1, \dots, x_n$  is a total order on the monomials which respects multiplication and has the unit 1 as the minimal element. A complete classification of admissible monomial orderings was first given in [11]. We will perform complexity analyses for a subclass of these orders, namely those defined by  $n \times n$ -matrices of rational numbers which we define below. This is a proper restriction, since given an admissible monomial order and a natural number, there is an  $n \times n$ -matrix of rational numbers that agrees with the given monomial order on all monomials whose degrees are bounded by the given number [8]. In the rest of the paper we will assume that all monomial orders are admissible.

Let  $A = (a_{ij})$  be an element in  $GL_n(\mathbb{Q})$  with the property that the first nonzero entry in each column is positive. Then we can induce an order  $\prec_A$  on the monomials in  $S$  by

$$x^\alpha \prec_A x^\beta \text{ iff } A\alpha^t < A\beta^t,$$

where  $A\alpha^t < A\beta^t$  is the lexicographic order on  $\mathbb{Z}^n$ , that is,  $(v_1, \dots, v_n) < (w_1, \dots, w_n)$  if  $v_1 = w_1, \dots, v_{i-1} = w_{i-1}$  and  $v_i < w_i$  for some  $i$ . Notice that multiplying a row of  $A$  by a positive integer does not change the order induced by  $A$ , hence we may assume that  $A$  is an integer matrix.

Given  $A$  and a monomial  $x^\alpha$ , we call the vector  $A\alpha^t$  the associated order vector to  $x^\alpha$  and we denote it by  $\text{ov}_A(x^\alpha)$ . The simple observation

$$\text{ov}_A(x_i x^\alpha) = A(\alpha_1, \dots, \alpha_{i-1}, \alpha_i + 1, \alpha_{i+1}, \dots, \alpha_n)^t = \text{ov}_A(x^\alpha) + (a_{1i}, \dots, a_{ni}),$$

gives us a handy formula for computing the order vector recursively.

An important subclass of the orders defined above consists of (1) the lexicographical order, (2) the degree lexicographical order and (3) the degree reverse lexicographical order. These orders, called *standard* in the rest of the paper, are the common most used ones in computer algebra and computer algebra systems have them predefined.

If  $x_1 \succ \dots \succ x_n$  for a standard order  $\prec$ , we have  $ov_{lex}(x^\alpha) = \alpha$ ,  $ov_{deglex}(x^\alpha) = (\sum_i \alpha_i, \alpha_1, \dots, \alpha_{n-1})$  and  $ov_{degrevlex} = (\sum_i \alpha_i, -\alpha_n, \dots, -\alpha_2)$ . It is easily seen how to compute  $ov(x_i x^\alpha)$  given  $ov(x^\alpha)$  for these orders.

In general, if  $x_{i_1} \succ \dots \succ x_{i_n}$  and  $\prec$  is standard, we will assume that the sequence  $i_1, \dots, i_n$  is given a priori and that  $x^\alpha$  means  $x_{i_1}^{\alpha_1} \dots x_{i_n}^{\alpha_n}$ . We do not make this assumption when  $\prec$  is given by a matrix. As indicated in the introduction, we will see that the complexity analysis of the monomial comparisons is dependent on whether the order is standard or not.

When  $\prec$  is a monomial order on  $x_1, \dots, x_n$  it will be useful to restrict  $\prec$  to a subset of the variables. If Ess is such a subset, we write  $\prec_{Ess}$  to denote the restriction of  $\prec$  to Ess.

### 2.3 Comparing Vectors of Integers

Since we assume that comparing two monomials is the same as lexicographically comparing their associated order vectors, we will now focus on comparing vectors of integers.

To be able to prove the next lemma, recall that the number of bits needed to represent an integer  $a$  is

$$\text{numbits}(a) = \begin{cases} 2 & \text{if } a = 0 \\ \lfloor \log_2(|a|) \rfloor + 2 & \text{otherwise} \end{cases} ,$$

where one bit is used to represent sign.

**Lemma 1.** *Let  $\alpha = (\alpha_1, \dots, \alpha_n)$  be a vector of integers  $\alpha_i$ . Let  $m = \sum_i |\alpha_i|$ . Then*

$$\sum_i \text{numbits}(\alpha_i) \leq \begin{cases} 3n & \text{if } m \leq 2n \\ n \log_2(m/n) + 2n & \text{otherwise} \end{cases}$$

*Proof.* Suppose that  $\alpha$  contains  $k$  non-zero entries. Without loss of generality, we may assume that  $\alpha_i = 0$  if  $i > k$  and  $\alpha_i \neq 0$  if  $i \leq k$ . We get

$$\begin{aligned} \sum_j \text{numbits}(\alpha_j) &= \sum_{j=1}^k \text{numbits}(\alpha_j) + \sum_{j=k+1}^n \text{numbits}(0) \\ &= \sum_{j=1}^k \lfloor \log_2(\alpha_j) \rfloor + 2k + 2(n - k) \leq \sum_{j=1}^k \log_2(\alpha_j) + 2n. \end{aligned}$$

Now  $\sum_{j=1}^k \log_2(\alpha_j) = \log_2(\alpha_1 \dots \alpha_k)$  and since  $\alpha_1 \dots \alpha_k \leq \underbrace{m/k \dots m/k}_{k \text{ times}}$ , we

conclude that  $\sum_j \text{numbits}(\alpha_j) \leq f(k)$  where  $f(k) = k \log_2(m/k) + 2n$ . We see that  $f'(k) = 0$  for  $k = m/2$  and that  $f(m/2)$  is a maximum. If  $m \leq 2n$ , then  $\sum_i (\text{numbits}(\alpha_i)) < f(m/2) = m/2 + 2n \leq 3n$ . If  $2n < m$ , then observe that  $f(k)$  is a monotone increasing function on the interval  $[1, m/2]$  so that  $\sum_i (\text{numbits}(\alpha_i)) < f(n) = n \log_2(m/n) + 2n$ .

We can formulate Lemma 1 in a more compact way by saying that the number of bits needed is proportional to  $n \max(\log(m/n), 1)$ .

**Lemma 2.** *Let  $v = (v_1, \dots, v_n)$  and  $w = (w_1, \dots, w_n)$  be vectors of integers such that  $\sum_i |v_i| = m_1$  and  $\sum_i |w_i| = m_2$ . Let  $m = \max(m_1, m_2)$ . In time  $O(n \max(\log(m/n), 1))$  we can determine if  $v$  and  $w$  differ and if they do, we can determine the index  $i$  where they differ.*

*Proof.* We proceed as follows. Compare  $v_1$  and  $w_1$ . If they differ, we stop and return the index. Otherwise, continue until we reach an index  $i$  such that  $v_i \neq w_i$  or  $v = w$ . Let us now analyze the complexity of this procedure. To compare the vectors  $v$  and  $w$ , we need to compare at most all entries, that is, we get time complexity proportional to  $\sum_i (\text{numbits}(v_i) + \text{numbits}(w_i))$ . We rewrite this sum as  $\sum_i \text{numbits}(v_i) + \sum_i \text{numbits}(w_i)$  and use Lemma 1 and the remark thereafter to conclude that this sum is dominated by  $n(\max(\log(m_1/n), 1) + \max(\log(m_2/n), 1))$ .

## 2.4 Comparing Monomials with Respect to a Standard Order

**Lemma 3.** *Suppose that  $\prec$  is a standard order. An upper bound for the time needed to compute  $\text{ov}(x_i x^\alpha)$  given  $\text{ov}(x^\alpha)$  is  $O(\log(m))$ , where  $m$  is the degree of  $x^\alpha$ .*

*Proof.* We need to increment at most two entries (in case of a degree order). The lemma follows since all entries are bounded by  $m$ .

**Lemma 4.** *An upper bound for the time needed to compare two monomials  $x^\alpha$  and  $x^\beta$  given  $\text{ov}_\prec(x^\alpha)$  and  $\text{ov}_\prec(x^\beta)$  with respect to a standard order is  $O(n \max(\log(m/n), 1))$ , where  $m = \max(\sum_i \alpha_i, \sum_i \beta_i)$ .*

*Proof.* Immediately from Lemma 2 when  $\prec$  is lex. When  $\prec$  is one of the degree orders, the first entry equals the degree and the sum of the rest of the entries is bounded by  $m$ , thus we get the same complexity in this case as well.

**Lemma 5.** *Let  $\prec$  be a standard monomial order and let  $\text{Ess} = \{x_{j_1}, \dots, x_{j_{\overline{n}}}\}$  be a subset of the variables such that  $x_{j_1} \succ \dots \succ x_{j_{\overline{n}}}$ . To compute  $\prec_{\text{Ess}}$  is  $O(1)$ .*

*Proof.* Immediate.

## 2.5 Comparing Monomials with Respect to a Matrix Order

**Lemma 6.** *An upper bound for the time needed to compute  $\text{ov}_A(x_i x^\alpha)$  given  $\text{ov}_A(x^\alpha)$  is proportional to  $n \log(cm)$ , where  $c = \max(|a_{ij}|)$  and  $m = \sum_i \alpha_i$ .*

*Proof.* From the recursion formulas given in section 2.2, we see that  $\text{ov}_A(x^\alpha \cdot x_i) = \text{ov}_A(x^\alpha) + (a_{1i}, \dots, a_{ni})$ , that is, we need to do  $n$  summations of integers bounded by  $cm$ .

**Lemma 7.** *An upper bound for the time needed to compare two monomials  $x^\alpha$  and  $x^\beta$  with respect to a matrix order defined by  $A$  given  $\text{ov}_A(x^\alpha)$  and  $\text{ov}_A(x^\beta)$  is proportional to  $n \log(cm)$ , where  $c = \max(|a_{ij}|)$  and  $m = \sum_i \alpha_i$ .*

*Proof.* We have  $n$  comparisons of integers bounded by  $cm$ .

Let  $\prec$  be a monomial order given by a matrix and let  $\text{Varord}_\prec(n)$  be the cost of determining  $i_1, \dots, i_n$  such that  $x_{i_1} \succ \dots \succ x_{i_n}$ . When  $\prec$  is a standard order we assume that  $i_1, \dots, i_n$  was given as input, so that  $\text{Varord}_\prec(n)$  is  $O(1)$ . However, we do not assume this for a general order given by a matrix. Instead we have

**Lemma 8.** *For an order given by a matrix, we can compute  $\text{Varord}_\prec(n)$  in time  $O(n^2 \log(c) \log(n))$ .*

*Proof.* To compare  $x_i \prec x_j$  is the same as comparing the  $i$ 'th and the  $j$ 'th column of the matrix  $A$  defining  $\prec$ . An upper bound for the comparisons is thus  $O(n \log(nc/n) + n) = O(n \log(c))$ . Since sorting is  $O(n \log(n))$  comparisons, the upper bound becomes  $O(n^2 \log(c) \log(n))$ .

**Lemma 9.** *Let  $\prec$  be a monomial order given by a matrix  $A$  and let  $\text{Ess} = \{x_{j_1}, \dots, x_{j_{\bar{n}}}\}$  be a subset of the variables such that  $x_{j_1} \succ \dots \succ x_{j_{\bar{n}}}$ . To determine an  $\bar{n} \times \bar{n}$ -matrix  $A_{\text{Ess}}$  such that  $\prec_{\text{Ess}}$  is given by  $A_{\text{Ess}}$  can be done using  $O(n\bar{n}^2)$  arithmetic operations over  $\mathbb{Q}$ .*

*Proof.* Clearly the  $n \times \bar{n}$  matrix  $\bar{A}$  obtained by keeping the columns  $j_1, \dots, j_{\bar{n}}$  defines  $\prec_{\text{Ess}}$  and it has rank  $\bar{n}$ . Suppose that the  $i$ 'th row of  $\bar{A}$  can be written as a linear combination of the rows whose indices are less than  $i$ . Let  $\alpha$  and  $\beta$  be two order vectors with respect to  $\text{Ess}$ . Then, if  $\bar{A}\alpha$  and  $\bar{A}\beta$  agree on the first  $i - 1$  rows, then they also agree on the  $i$ 'th row. Hence the  $i$ 'th row is superfluous. Thus, to determine  $A_{\text{Ess}}$  is the same as rowreducing  $\bar{A}$ , which has arithmetic complexity  $O(n\bar{n}^2)$ .

## 2.6 Merging Sorted Lists of Monomials

Let  $v = (v_1, \dots, v_n)$  and  $w = (w_1, \dots, w_n)$  be two  $n$ -vectors of non-negative integers. If  $v \neq w$ , let  $\Delta(v, w)$  be the first index where  $v$  and  $w$  differ. If  $v = w$  let  $\Delta(v, w) = n + 1$ .

**Lemma 10.** *If  $u < v$ ,  $u < w$  and  $\Delta(u, w) < \Delta(u, v)$ , then  $u < v < w$  and  $\Delta(v, w) = \Delta(u, w)$ .*

*Proof.* Let  $k = \Delta(u, w)$ . Then  $u_1 = w_1, \dots, u_{k-1} = w_{k-1}$  and  $u_k < w_k$ . Since  $k < \Delta(u, v)$ , we have  $u_1 = v_1, \dots, u_k = v_k$ . Hence  $v_1 = w_1, \dots, v_{k-1} = w_{k-1}$  and  $v_k < w_k$  and thus  $v < w$  and  $\Delta(v, w) = k = \Delta(u, w)$ .

**Lemma 11.** *If  $u < v$ ,  $u < w$  then  $\Delta(v, w) \geq \min(\Delta(u, v), \Delta(u, w))$ .*

*Proof.* If  $\Delta(u, w) < \Delta(u, v)$  or  $\Delta(u, v) < \Delta(u, w)$ , then the lemma follows by Lemma 10. Otherwise,  $\Delta(u, v) = \Delta(u, w)$  implies that  $v$  and  $w$  agree on the first  $\Delta(u, v)$  positions.

**Lemma 12.** *Let  $a = (a_1, \dots, a_t)$  be a list of  $n$ -tuples of elements in an ordered set  $\Sigma$ . Suppose that  $a$  is sorted lexicographically in increasing order and that we are given  $\Delta(a_i, a_{i+1})$  for  $i = 1, \dots, t - 1$ . Let  $b$  be any element in  $\Sigma^n$ . Using  $O(t + n)$  comparisons of elements in  $\Sigma$  plus time proportional to  $t \log(n)$ , we can find an index  $i$ ,  $0 \leq i \leq t$ , such that  $a_1 \leq \dots \leq a_i < b \leq a_{i+1} \leq \dots \leq a_t$  and  $\Delta(a_i, b)$  when  $i \geq 1$  and  $\Delta(b, a_{i+1})$  when  $i < t$ .*

*When  $\Sigma$  is the set of non-negative integers and  $\sum_i v_i \leq m$  for all  $v \in a$  and  $\sum_i b_i \leq m$ , an upper bound for the time complexity is  $O(n \max(\log(m/n), 1) + t \log(\max(n, m)))$ .*

*Proof.* The proof contains three parts. We (a) give an algorithm, (b) prove its correctness and (c) show that the complexity of the algorithm agrees with what was stated in the lemma.

### The algorithm

At stage 0, compute  $k = \Delta(a_1, b)$ . If  $a_1 < b$ , continue with stage 1. If  $b \leq a_1$  we stop and return 0 and  $\Delta(a_1, b) = k$ .

At stage  $i$ ,  $1 \leq i < t$ , we suppose that  $a_i < b$  and that  $\Delta(a_i, b)$  is computed.

We consider four different cases.

If  $a_i = a_{i+1}$

Clearly  $\Delta(a_{i+1}, b) = \Delta(a_i, b)$ . Continue with stage  $i + 1$ .

Else, if  $\Delta(a_i, b) > \Delta(a_i, a_{i+1})$

We have  $a_i < b < a_{i+1}$  and  $\Delta(a_{i+1}, b) = \Delta(a_i, a_{i+1})$  by Lemma 10.

Thus, we stop and return  $i$ ,  $\Delta(a_i, b)$  and  $\Delta(a_{i+1}, b)$ .

Else, if  $\Delta(a_i, b) < \Delta(a_i, a_{i+1})$

We have  $a_i < a_{i+1} < b$  and  $\Delta(a_{i+1}, b) = \Delta(a_i, b)$ , again by Lemma 10.

Continue with stage  $i + 1$ .

Else

We have  $\Delta(a_i, b) = \Delta(a_i, a_{i+1}) = k$ . Compare  $b_k$  and  $a_{i+1, k}$ ,  $b_{k+1}$  and  $a_{i+1, k+1}$  and so on to find  $k' = \Delta(a_{i+1}, b)$ .

If  $k' = n + 1$

We have  $a_i < b = a_{i+1}$ , so we stop and return  $i$ ,  $\Delta(a_i, b)$  and  $\Delta(a_{i+1}, b)$ .

Else, if  $b_{k'} < a_{i+1, k'}$

We have  $a_i < b < a_{i+1}$ , so we stop and return  $i$ ,  $\Delta(a_i, b)$  and  $\Delta(a_{i+1}, b)$ .

Else

We have  $b_{k'} > a_{i+1, k'}$ , so  $a_i < a_{i+1} < b$ . Continue with stage  $i + 1$ .

At stage  $t$  we have  $a_t < b$ . Thus, we stop and return  $t$  and  $\Delta(a_t, b)$ .

### The correctness of the algorithm

By construction.

### Complexity of the algorithm

There are two key indices that we update during the algorithm. The first ( $i$ ) refers to a position in the list  $a$ , the second ( $k$ ) refers to a position in the vector

$b$ . Notice that after each comparison, either  $i$  or  $k$  is increasing. Since both  $i$  and  $k$  are non-decreasing it follows that the number of comparisons of elements in  $\Sigma$  is at most  $n + t$ .

The number of  $\Delta$ -comparisons during the algorithm is one per stage, that is, at most  $t$ . Every such comparison consists of comparing integers bounded by  $n$ . We conclude that the time needed for the integer comparisons is proportional to  $t \log(n)$ .

Let now  $\Sigma$  be the set of non-negative integers. Everytime we increase  $i$ , we make a comparison of an integer bounded by  $m$ , this gives time proportional to  $t \log(m)$ . However, when increasing  $k$ , we are in a situation where  $a_{ik} = b_k$ . Hence, the total time used for the increasings of  $k$  is proportional to  $n \max(\log(m/n), 1)$  by Lemma 2. Only once during the algorithm we will compare  $a_{ik}$  and  $b_k$  to conclude that they differ, this cost is  $O(\log(m))$ . It follows that an upper bound for the algorithm is proportional to

$$\begin{aligned} t \log(n) + t \log(m) + n \max(\log(m/n), 1) + \log(m) \\ = t \log(\max(m, n)) + n \max(\log(m/n), 1). \end{aligned}$$

**Theorem 1.** *Let  $a = (a_1, \dots, a_t)$  and  $b = (b_1, \dots, b_s)$  be two lists of  $n$ -tuples of elements in an ordered set  $\Sigma$ . Suppose that  $a$  and  $b$  are sorted lexicographically with respect to the order  $<$  in  $\Sigma$ . Suppose that we are given  $\Delta(a_i, a_{i+1})$  for  $i = 1, \dots, t - 1$  and  $\Delta(b_i, b_{i+1})$  for  $i = 1, \dots, s - 1$ . We can merge  $a$  and  $b$  into a new list  $c$  and compute the sequence  $\Delta(c_1, c_2), \Delta(c_2, c_3), \dots$  using  $O(sn + t)$  comparisons plus time complexity  $O(t \log(n))$ . When  $\Sigma$  is the non-negative integers and  $\sum_i v_i \leq m$  for all  $v \in a \cup b$ , an upper bound for the time complexity of the algorithm is  $O(\min(s, t)n \max(\log(m/n), 1) + \max(s, t) \log(n + m))$ .*

*Proof.* Suppose, without loss of generality, that  $s < t$ . Let  $i_1$  be the index returned after calling the algorithm in Lemma 12 with  $a$  and  $b_1$ . Without affecting the complexity of the algorithm, it is clear that we can modify it to give a list  $c = (a_1, \dots, a_{i_1}, b_1)$ . Suppose that  $i_2$  is the index returned after calling the algorithm in Lemma 12 with  $a_{i_1+1}, \dots, a_t$  and  $b_2$ . Again, it is clear that it is possible to update the list  $c$  without affecting the complexity so that it reads  $(a_1, \dots, a_{i_1}, b_1, a_{i_1+1}, \dots, a_{i_2}, b_2)$ . If we proceed in this way we obtain the sequence

$$a_1 \leq \dots \leq a_{i_1} < b_1 \leq a_{i_1+1} \leq \dots \leq a_{i_2} < b_2 \leq \dots < b_s \leq a_{i_s+1} \leq \dots \leq a_t.$$

Since the algorithm in Lemma 12 returns  $\Delta(b_j, a_{i_j+1})$  we only need to check the case when  $a_{i_j} = a_{i_j+1}$  in order to conclude that  $\Delta(c_1, c_2), \Delta(c_2, c_3), \dots$  is computed as a side effect of the calls to the modified algorithm. But when  $a_{i_j} = a_{i_j+1}$  we have that  $b_j$  and  $b_{j+1}$  are consecutive and hence  $\Delta(b_j, b_{j+1})$  is already computed by assumption.

Although it does not affect the complexity, stage 0 of the algorithm given in Lemma 12 can be modified. When calling with  $a_{i_j+1}, \dots, a_t$  and  $b_{j+1}$  we can use that  $\Delta(b_j, a_{i_j+1})$  already is computed. Indeed, by Lemma 11,  $\Delta(b_{j+1}, a_{i_j+1}) \geq$

$\min(\Delta(b_j, a_{i_j+1}), \Delta(b_j, b_{j+1}))$ , so we could call the algorithm in Lemma 12 with the extra parameter  $\min(\Delta(b_j, a_{i_j+1}), \Delta(b_j, b_{j+1}))$  to speed up the computation of  $\Delta(b_{j+1}, a_{i_j+1})$  in stage 0.

Since we make  $s$  calls to the algorithm in Lemma 12 (or to be more precise, to the modified algorithm as defined above), we make  $((i_1 + 1) + n) + ((i_2 + 1 - (i_1 + 1) + n) + \dots + ((i_s + 1 - (i_{s-1} + 1)) + n) = (i_s + 1) + sn < t + sn$  comparisons of elements in  $\Sigma$ . The time complexity for the integer comparison part is by Lemma 12 proportional to  $(i_1 + 1)\log(n) + (i_2 + 1 - (i_1 + 1))\log(n) + \dots + (i_s + 1 - (i_{s-1} + 1))\log(n) = (i_s + 1)\log(n) < t\log(n)$ . If  $\Sigma$  is the non-negative integers, then an upper bound becomes  $O(sn \max(\log(m/n), 1) + t\log(n + m))$  by Lemma 2.

We now give two applications of the new merge algorithm. The idea of the first example is to make the algorithm clear to the reader, while the second example shows the strength of the algorithm.

*Example 1.* Suppose that we want to merge the lists

$$a = (x_1x_3^2x_4^2, x_1x_3^3, x_1^2x_4, x_1^2x_2x_5, x_1^2x_2x_4^2x_5, x_1^3)$$

and

$$b = (x_1, x_1x_3^2, x_1^2x_2x_4x_5, x_1^2x_2x_4^2x_5)$$

of monomials, sorted in increasing order with respect to lex and  $x_1 > x_2 > x_3 > x_4 > x_5$ . Since  $ov_{lex}(x^\alpha) = \alpha$ , we will use the exponent vectors. Thus, in accordance with the notation above,  $a_1 = (1, 0, 2, 2, 0), a_2 = (1, 0, 3, 0, 0), a_3 = (2, 0, 0, 1, 0), a_4 = (2, 1, 0, 0, 1), a_5 = (2, 1, 0, 2, 1), a_6 = (3, 0, 0, 0, 0)$  and  $b_1 = (1, 0, 0, 0, 0), b_2 = (1, 0, 2, 0, 0), b_3 = (2, 1, 0, 1, 1), b_4 = (2, 1, 0, 2, 1)$ .

We begin by comparing  $b_1$  and  $a_1$ . We see that  $\Delta(b_1, a_1) = 3$  and that  $b_{13} < a_{13}$ , hence  $b_1 < a_1$ . Since  $\Delta(b_1, b_2) = 3$ , we compare  $b_{23}$  and  $a_{13}$ . They are equal, so we check the fourth index and conclude that  $b_2 < a_1$ . Now  $\Delta(b_2, b_3) = 1$ , so we conclude that  $a_1 < b_3$  only by checking the first index. Since  $\Delta(a_1, a_2) = 3 > 1$ , we also have  $a_2 < b_3$ . But  $\Delta(a_2, a_3) = 1$ , so we compare  $a_3$  and  $b_3$  from index 1 and conclude that  $a_3 < b_3$  and  $\Delta(a_3, b_3) = 2$ . We see that  $\Delta(a_3, a_4) = 2$ , thus we compare  $b_{32}$  and  $a_{42}$ . They are equal, so are  $b_{33}$  and  $b_{43}$ , but  $b_{34} > a_{44}$ , hence  $a_4 < b_3$  and  $\Delta(b_3, a_4) = 4$ . Since also  $\Delta(a_4, a_5) = 4$ , we compare the fourth index of  $b_3$  and  $a_5$  to conclude that  $b_3 < a_5$ . Finally, since  $\Delta(b_3, b_4) = 4$ , we check the fourth and fifth indices of  $b_4$  and  $a_5$  to conclude that  $b_4 = a_5$ . We have

$$b_1 < b_2 < a_1 < a_2 < a_3 < a_4 < b_3 < b_4 \leq a_5 < a_6.$$

and the sequence of differences is

$$3, 4, 2, 1, 2, 4, 4, 6, 1.$$

*Example 2.* Let  $n = 2s$ . Let  $f = x_1x_3 + x_2x_s$  and let  $g = x_1x_2 + x_2x_3 + \dots + x_{n-1}x_n$ . Let  $\prec$  be degrevlex with respect to  $x_1 \succ \dots \succ x_n$ . We see that the terms of  $f$  and  $g$  are written with respect to this order. Suppose that, during

a Grbner basis computation, we want to compute the S-polynomial of  $f$  and  $g$ , that is, we want a sorted expression of  $S(f, g) = x_2f - x_3g$ . This is the same as merging  $x_2^2x_s$  and  $(-x_2x_3^2, -x_3^2x_4, -x_3x_4x_5, -x_3x_5x_6, \dots, -x_3x_{n-1}x_n)$  (together with an arithmetic operation in the case of equality). For simplicity, we write these expressions as lists of order vectors and omit the coefficients. We get

$$a = (3, \underbrace{0, \dots, 0}_{s \text{ times}}, -1, 0, \dots, 0, -2)$$

and

$$b = ((3, 0, \dots, 0, -2, -1), (3, 0, \dots, 0, -1, -2, 0), (3, 0, \dots, 0, -1, -1, -1, 0), (3, 0, \dots, 0, -1, -1, 0, -1, 0), \dots, (3, -1, -1, 0, \dots, 0, -1, 0)).$$

We assume that we are given the sequence of differences for  $g$  a priori. Since the sequence of differences is closed under multiplication with a monomial, we obtain the sequence of differences for  $b$ . It reads  $(n - 1, n - 2, \dots, 2)$ . Using the algorithm in Lemma 12, we first compare  $a$  and  $b_1$ . After  $s + 2$  comparisons, we see that  $a \succ b_1$ . Since  $\Delta(b_1, b_2) = n - 2$  and  $\Delta(a, b_1) = s + 2$ , we get (after comparing  $n - 2$  and  $s + 2$ ) that  $a \succ b_2$  and that  $\Delta(a, b_2) = s + 2$ . Continuing this way we see that  $a \prec b_1, a \prec b_2, a \prec b_3, \dots, a \prec b_{s-3}$ , and  $\Delta(a, b_i) = s + 2$  for  $i = 1, \dots, s - 3$ , using

$$s + 2 + \underbrace{1 + \dots + 1}_{s-3 \text{ times}} = 2s - 1$$

comparisons. We have  $\Delta(b_{s-3}, b_{s-2}) = s + 2$  and since also  $\Delta(a, b_{s-3}) = s + 2$ , we compare  $b_{s-2, s+2}$  and  $a_{s+2}$  to conclude that  $a \succ b_{s-2}$ . In total, we have used  $2s - 1 + 2 = 2s + 1$  comparisons.

Proceeding in a naive way we would use  $(s + 2) \cdot (s - 2) = s^2 - 4$  comparisons. It should be remarked that one needs extra cost for the bookkeeping of  $\Delta(a, b_i)$ , for  $i = 1, \dots, s - 2$ , which is of the same magnitude as the comparisons, that is, to be fair, we should compare  $2s + 1 + s - 2 = 3s - 1$  with  $s^2 - 4$ .

### 3 The Buchberger-Möller -Algorithm Revised

We first fix some notation. If  $x^\alpha$  is a monomial, we define its support, which we denote by  $\text{supp}(x^\alpha)$ , to be the set of all  $x_i$  such that  $\alpha_i > 0$ . If  $M$  is a set of monomials, then we define the support of  $M$  to be the union of the supports of the elements in  $M$ . Let  $I$  be an ideal in  $S$  and let  $B$  be any subset of  $S$  such that  $[B] = \{[b] : b \in B\}$  is a vector space basis for  $S/I$ . Here  $[b]$  denotes the equivalence class in  $S/I$  containing  $b$ . If  $s$  is an element in  $S$ , its residue can be uniquely expressed as a linear combination of the elements in  $[B]$ , say  $[s] = \sum c_i [b_i]$ . The  $S$ -element  $\sum c_i b_i$  is then called the normal form of  $s$  with respect to  $B$  and we write  $\text{Nf}(s, B) = \sum c_i b_i$ . We abuse notation and say that  $B$  (instead of  $[B]$ ) is a basis for  $S/I$ .

Let  $\prec$  be an (admissible) monomial order. The initial ideal of  $I$ , denoted by  $\text{in}(I)$ , is the monomial ideal consisting of all leading monomials of  $I$  with respect to  $\prec$ . One of the characterizations of a set  $G$  being a Grbner basis of an ideal  $I$  with respect to a monomial order  $\prec$  is that  $G \subseteq I$  and that the leading terms of  $G$  generate  $\text{in}(I)$ . An old theorem by Macaulay states that the residues of the monomials outside  $\text{in}(I)$  form a  $\mathbb{k}$ -basis for the quotient  $S/I$ . The set of monomials outside  $\text{in}(I)$  is closed under taking submonomials.

If  $p$  is a point in  $\mathbb{k}^n$  and  $f$  is an element of  $S$ , we denote by  $f(p)$  the evaluation of  $f$  at  $p$ . When  $P = \{p_1, \dots, p_m\}$  is a set of points,  $f(P) = (f(p_1), \dots, f(p_m))$ . If  $F = \{f_1, \dots, f_s\}$  is a set of elements in  $S$ , then  $F(P)$  is defined to be the  $s \times m$  matrix whose  $i$ 'th row is  $f_i(P)$ .

The vanishing ideal  $I(P)$  is the ideal consisting of all elements in  $S$  which vanishes on all the points in  $P$ . If  $f_1$  and  $f_2$  are two elements in  $S$  and  $[f_1] = [f_2]$  in  $S/I(P)$ , then  $f_1(p) = f_2(p)$  for  $p \in P$ . That a set  $[B]$  of  $m$  elements is a  $\mathbb{k}$ -basis for  $S/I(P)$  is equivalent to  $\dim_{\mathbb{k}}(B(P)) = m$ .

In the rest of the paper, we will refer to the Buchberger-Möller algorithm as the “BM-algorithm”. The BM-algorithm takes as input a set of points in  $\mathbb{k}^n$  and a monomial order. It returns a Grbner basis  $G$  of  $I$  and the set  $B$  of monomials outside  $\text{in}(I)$ . The BM-algorithm was first given in [3]. During the years it has been reformulated and modified. In the paper [4], the ideas of the BM-algorithm was used to switch between different Grbner bases of a zero-dimensional ideal. In the unifying paper [10] it was shown that both the BM- and the FGLM-algorithm can be seen as an algorithm that computes a Grbner basis from an ideal defined by functionals. The complexity studies given in [10] apply to the BM-algorithm and in fact, [10] is by tradition the paper which one refers to when complexity issues of the BM-algorithm are discussed. However, most of the complexity studies in [10] are done by referring to the paper [4].

We will first discuss the complexity studies of the BM-algorithm and postpone the connection with ideals defined by functionals to section 3.3.

### 3.1 Two Formulations of the BM-Algorithm

We give below the formulation of the BM-algorithm as given in [1]. When the algorithm terminates,  $G$  is the Grbner basis with respect to  $\prec$  and  $B$  is the complement of the initial ideal with respect to  $\prec$ .

- C1.** Start with empty lists  $G = B = R = [ ]$  a list  $L = [1]$ , and a matrix  $C = (c_{ij})$  over  $\mathbb{k}$  with  $m$  columns and initially zero number of rows.
- C2.** If  $L = [ ]$ , return the pair  $[G, B]$  and stop. Otherwise, choose the monomial  $t = \min_{\prec}(L)$ , the smallest according to the ordering  $\prec$ . Delete  $t$  from  $L$ .
- C3.** Compute the evaluation vector  $(t(p_1), \dots, t(p_m)) \in \mathbb{k}^m$ , and reduce it against the rows of  $C$  to obtain

$$(v_1, \dots, v_m) = (t(p_1), \dots, t(p_m)) - \sum_i a_i(c_{i1}, \dots, c_{im}) \quad a_i \in \mathbb{k}.$$

- C4.** If  $(v_1, \dots, v_m) = (0, \dots, 0)$ , then append the polynomial  $t - \sum_i a_i r_i$  to the list  $G$ , where  $r_i$  is the  $i$ 'th element of  $R$ . Continue with step C2. 1
- C5.** If  $(v_1, \dots, v_m) \neq (0, \dots, 0)$ , then add  $(v_1, \dots, v_m)$  as a new row to  $C$ , and  $t - \sum_i a_i r_i$  as a new element to  $R$ . Append the power product  $t$  to  $B$ , and add to  $L$  those elements of  $\{x_1 t, \dots, x_n t\}$  which are neither multiples of an element of  $L$  nor of  $\text{in}(G)$ . Continue with step C2.

The authors in [1] claims that this is the same as the algorithm restricted to the BM-situation which appeared in [10], but this is not exactly the case. To get the algorithm given in [10] restricted to the BM-situation using the five-step description, we need to reformulate steps 2 and 5.

- C2'.** If  $L = [ ]$ , return the pair  $[G, B]$  and stop. Otherwise, choose the power product  $t = \min_{\prec}(L)$ , the smallest according to the ordering  $\prec$ . Delete  $t$  from  $L$ . If  $t$  is a multiple of an element in  $\text{in}(G)$ , then repeat this step. Else, continue with step C3.
- C5'.** If  $(v_1, \dots, v_m) \neq (0, \dots, 0)$ , then add  $(v_1, \dots, v_m)$  as a new row to  $C$ , and  $t - \sum_i a_i r_i$  as a new element to  $R$ . Append the power product  $t$  to  $B$ , and merge  $\{x_1 t, \dots, x_n t\}$  and  $L$ . Continue with step C2'.

It is an easy exercise to show that the output of the two algorithms agree and that the two variants give the same arithmetic complexity, which is reported [1,4,10] to be  $O(nm^3)$  arithmetic operations. We now give a better bound.

**Proposition 1.** *The arithmetic complexity of the algorithms given above agree and an upper bound is  $O(nm^2 + \min(m, n)m^3)$ .*

*Proof.* The arithmetic operations are performed in steps C3 and C4. Step C3 involves an evaluation and a row reduction. To compute  $(t(p_1), \dots, t(p_m))$  requires  $m$  multiplications, since  $t = x_i t'$  for some  $x_i$  and some  $t'$  and  $t'(p_1), \dots, t'(p_m)$  already has been evaluated. The row reduction requires  $O(m^2)$  arithmetic operations. Step C4 consist of expressing  $\sum_i a_i r_i$  in the basis and is also an  $O(m^2)$  operation. Notice that the number of calls to C4 is exactly  $|B| = m$ , the number of calls to C5/C5' is exactly  $|G|$  and thus, the number of calls to C3 is exactly  $m + |G|$ . Thus, the number of arithmetic operations is proportional to  $(m + |G|) \cdot m^2$ , so it is enough to prove that  $|G|$  is proportional to  $n + \min(m, n)m$ .

The number of variables in  $B$  is at most  $\min(n, m - 1)$  since the basis consist of  $m$  elements and  $1 \in B$ . Accordingly, if we let  $s$  denote the number of variables in  $\text{in}(G)$ , then  $n - \min(n, m - 1) \leq s \leq n$  ( $s = n$  only if  $m = 1$ ). Now notice that if  $x_i t$  is an element inserted into  $L$ , then, since  $x_i < x_i t$  and  $x_i$  is inserted into  $L$  during the first step of the algorithm,  $x_i$  will be treated before  $x_i t$ . This shows that when we add a new element  $t$  to  $B$  and  $\{x_1 t, \dots, x_n t\}$  to  $L$ , we know that at most  $n - s$  of these elements would be added to  $\text{in}(G)$ , because if  $x_i \in \text{in}(G)$ , then  $x_i t \in \text{in}(G)$ .

Since we add  $m$  elements to  $B$ , we see that the elements of degree more than one in  $\text{in}(G)$  is at most  $m(n - s)$ . We conclude that the number of elements in  $\text{in}(G)$  is at most  $m(n - s) + s < m \min(n, m - 1) + n$ .

The parts of the algorithms that concern monomial manipulations are harder to analyze. As stated in the introduction, the number of integer comparisons in the monomial manipulation-part is reported to be proportional to  $n^2m^2$ , assuming a standard order. We agree that this is an upper bound for the algorithm using steps C2' and C5'. Since it is not explained in [1] how to check if  $t$  is a multiple of an element in  $L$  or of  $\text{in}(G)$ , the actual behavior of this algorithm might be worse than the algorithm using step C2' and C5', although the list  $L$  during the former algorithm contains less elements than the list  $L$  during the latter.

To check if  $t$  is a multiple of an element in  $\text{in}(G)$  using step C2' and C5' is simple and is due to the following nice observation given in [4], pp 336.

**Lemma 13.** *Using steps C2' and C5', to check if  $t$  is a multiple of  $\text{in}(G)$  can be replaced by checking if  $|\text{supp}(t)| > \text{cp}(t)$ , where  $\text{cp}(t)$  denote the number of copies of  $t$  in  $L$ .*

We remark that this simplification of the check does not apply when using steps C2 and C5.

### 3.2 Optimizing the BM-Algorithm

In order to improve the time complexity during the monomial manipulations part, we will use a projection technique together with Theorem 1. The idea is to identify a set  $\text{Ess}$  of the variables with the property that  $\text{supp}(B) \subseteq \text{Ess}$  and  $|\text{Ess}| \leq \min(m - 1, n)$ . Once the set  $\text{Ess}$  is identified, we only consider monomials in the monoid generated by  $\text{Ess}$ , hence the associated order vectors for the monomials is of length bounded by  $\min(m - 1, n)$ .

The projection technique is covered in the following lemma and is in some sense inspired by [6,7].

**Lemma 14.** *Let  $\prec$  be a monomial order and let  $p_1, \dots, p_m$  be distinct points in  $\mathbb{k}^n$ . In  $O(nm^2)$  arithmetic operations plus time bounded by  $\text{Varord}_\prec(n)$ , we can determine a subset  $\text{Ess}$  of  $\{x_1, \dots, x_n\}$  with the following properties.*

- $\text{supp}(B) \subseteq \text{Ess}$
- $|\text{Ess}| \leq \min(m - 1, n)$
- $\dim_{\mathbb{k}}(\text{Ess}(P)) = |\text{Ess}|$  ( $\text{Ess}(P)$  has full rank.)

Moreover, for any element  $x_k \notin \text{Ess}$ , we obtain a “pseudo-normal form”  $x_k - \sum_j c_{kj} x_{i_j} \in I(P)$  where  $x_k \succ x_{i_j} \in \text{Ess}$  if  $c_{kj} \neq 0$ .

*Proof.* We first determine  $i_1, \dots, i_n$  such that  $x_{i_1} \succ \dots \succ x_{i_n}$  which has time complexity  $\text{Varord}_\prec(n)$ . Let

$$E_j = \begin{cases} \{\} & \text{if } j = 0 \\ E_{j-1} \cup \{x_{i_{n-j+1}}\} & \text{if } x_{i_{n-j+1}}(P) \notin \text{span}_{\mathbb{k}}\{1(P), E_{j-1}(P)\} \\ E_{j-1} & \text{otherwise} \end{cases}$$

and let  $\text{Ess} = E_n$ . It is clear that if  $x_k \notin \text{Ess}$ , then  $x_k \in \text{in}(I)$ , since  $x_k$  can be written as a linear combination of smaller elements, hence  $\text{supp}(B) \subseteq \text{Ess}$ .

By a dimension argument, we have  $|\text{Ess}| \leq m - 1$  and since  $\text{Ess}$  is a subset of the variables, clearly  $|\text{Ess}| \leq n$ . By construction  $\dim_{\mathbb{k}}(\text{Ess}(P)) = |\text{Ess}|$ . To determine if  $x_{i_{n-j+1}}(P) \notin \text{span}_{\mathbb{k}}\{1(P), E_{j-1}(P)\}$  is  $O(m^2)$  arithmetic operations by using a matrix representation. This is repeated  $n$  times, which gives the arithmetic complexity  $O(nm^2)$ . Finally, when  $x_k \notin \text{Ess}$ , we obtain an expression  $x_k = \sum_j c_{kj} x_{i_j} \pmod{I(P)}$  as a side effect of the matrix representation.

In the sequel, suppose that  $\text{Ess} = \{x_{i_1}, \dots, x_{i_{\overline{n}}}\}$ . Let  $\pi$  be the projection from  $\mathbb{k}^n$  to  $\mathbb{k}^{\overline{n}}$  with respect to  $\text{Ess}$ , that is,  $\pi((a_1, \dots, a_n)) = (a_{i_1}, \dots, a_{i_{\overline{n}}})$ . Let  $\pi^*$  be the corresponding monomorphism from  $T = \mathbb{k}[y_{i_1}, \dots, y_{i_{\overline{n}}}]$  to  $S$  given by  $y_{i_j} \mapsto x_{i_j}$ . If  $f$  is any element in  $T$ , then by construction  $\pi^*(f)(p) = f(\pi(p))$ . This shows that if  $\text{Ess}(P)$  has full rank, so has  $(\pi^*)^{-1}(\text{Ess})(\pi(P))$ . It follows that the points in  $\pi(P)$  are distinct. This leads us to use the following isomorphism result.

**Lemma 15.** *Let  $p_1, \dots, p_m$  be distinct points in  $\mathbb{k}^n$ . Let  $I$  be the vanishing ideal with respect to these points. Let  $\pi$  be a projection from  $\mathbb{k}^n$  to  $\mathbb{k}^{\overline{n}}$  such that  $\pi(p_1), \dots, \pi(p_m)$  are distinct. Let  $T = \mathbb{k}[y_{i_1}, \dots, y_{i_{\overline{n}}}]$  and let  $J$  be the vanishing ideal with respect to  $\pi(p_1), \dots, \pi(p_m)$ . Then  $S/I$  and  $T/J$  are isomorphic as algebras.*

*Proof.* Let  $\pi^*$  be the map defined above. For  $f \in T$  we have  $\pi^*(f)(p) = f(\pi(p))$ . Notice that  $f \in J$  is equivalent to  $f(\pi(q_i)) = 0, \forall i$ , which is equivalent to  $\pi^*(f)(p_i) = 0, \forall i$ , which is equivalent to  $\pi^*(f) \in I$ . This allows us to extend  $\pi^*$  to a monomorphism from  $T/J$  to  $S/I$ . Since  $\pi(p_1), \dots, \pi(p_m)$  are distinct, we have  $\dim_{\mathbb{k}}(T/J) = \dim_{\mathbb{k}}(S/I)$  and thus, the extension of  $\pi^*$  is an isomorphism of algebras.

**Lemma 16.** *Let  $\prec$  be a monomial order on  $S$ . Let  $\text{Ess}$  be a subset of the variables such that  $\text{supp}(B) \subseteq \text{Ess}$ ,  $|\text{Ess}| \leq \min(m - 1, n)$  and  $\dim_{\mathbb{k}}(\text{Ess}(P)) = |\text{Ess}|$ , where  $B$  is the monomials outside  $\text{in}(I(P))$  with respect to  $\prec$ . Let  $\pi$  be the projection defined by  $\pi((a_1, \dots, a_n)) = (a_{i_1}, \dots, a_{i_{\overline{n}}})$  and let  $\pi^*$  be the corresponding monomorphism from  $T = \mathbb{k}[y_{i_1}, \dots, y_{i_{\overline{n}}}]$  to  $S$ . Let  $\prec'$  be the monomial order defined by  $y^\alpha \prec' y^\beta$  if  $\pi^*(y^\alpha) \prec \pi^*(y^\beta)$ . Let  $B' = \{b'_1, \dots, b'_m\}$  be the set of monomials outside  $\text{in}(I(\pi(P)))$  with respect to  $\prec'$ . Then  $\pi^*(B') = B$ .*

*Proof.* Suppose that  $\pi^*(b'_i)$  can be written as a linear combination of elements in  $B$ ;  $\pi^*(b'_i) = \sum_j c_j b_j$  with  $\pi^*(b'_i) \succ b_j$  if  $c_j \neq 0$ . Since  $\text{supp}(B) \subseteq \text{Ess}$ , we get  $b'_i = \sum_j c_j (\pi^*)^{-1}(b_j)$  with  $b'_i \succ' (\pi^*)^{-1}(b_j)$ , which is a contradiction. Hence  $\pi^*(B') \subseteq B$  from which it follows that  $\pi^*(B') = B$  by Lemma 15 and a dimension argument.

**Lemma 17.** *In the context of Lemma 14 and Lemma 16, suppose that  $x_k - \sum_j c_{kj} x_{i_j} \in I(P)$  for all  $x_k$  outside  $\text{Ess}$ . Let  $G'$  be a reduced Grbner basis for  $I(\pi(P))$ . Then*

$$G = \pi^*(G') \sqcup \{x_k - \sum_j c_{kj} \pi^*(\text{Nf}((\pi^*)^{-1}(x_{i_j}), G'))\}$$

*is a reduced Grbner basis for  $I(P)$ .*

*Proof.* Since  $\pi^*(B')$  is the complement of  $\text{in}(I(P))$  by Lemma 16, it follows that  $\text{in}(I(P))$  is minimally generated by  $\text{Ess}^c \sqcup \pi^*(\text{in}(I(\pi(P))))$ . Clearly  $\pi^*(G')$  is contained in  $G$ . Thus, it is enough to prove that

$$x_k - \sum_j c_{kj} \pi^*(\text{Nf}((\pi^*)^{-1}(x_{i_j}), G')) \in I(P)$$

and that  $x_k$  is larger than any monomial occurring in the right hand sum. Since  $x_k - \sum_j c_{kj} x_{i_j} \in I(P)$ , we have that

$$x_k - \sum_j c_{kj} (\pi^*)(\text{Nf}((\pi^*)^{-1}(x_{i_j}), G')) \in I(P)$$

is equivalent to

$$\sum_j c_{kj} x_{i_j} - \sum_j c_{kj} (\pi^*)(\text{Nf}((\pi^*)^{-1}(x_{i_j}), G')) \in I(P).$$

Using the monomorphism  $(\pi^*)^{-1}$  we see that this is equivalent to

$$\sum_j c_{kj} ((\pi^*)^{-1}(x_{i_j}) - \text{Nf}((\pi^*)^{-1}(x_{i_j}), G')) \in I(\pi(P)).$$

Since each term  $(\pi^*)^{-1}(x_{i_j}) - \text{Nf}((\pi^*)^{-1}(x_{i_j}), G')$  is in  $G'$ , we are done with the first part. The second part follows since  $x_k \succ x_{i_j}$  if  $c_{kj} \neq 0$  and each  $\pi^*(x_{i_j})$  is written as a linear combination of elements less than  $\pi^*(x_{i_j})$  with respect to  $\prec'$ .

We now give an example of our method.

*Example 3.* Let  $p_1 = (1, 1, 0, 1, 0), p_2 = (2, 2, 1, 1, 1), p_3 = (2, 0, 1, 1, -1), p_4 = (5, 3, 4, 1, 2)$  be points in  $\mathbb{Q}^5$ . Let  $\prec$  be pure lex with respect to  $x_1 \succ x_2 \succ x_3 \succ x_4 \succ x_5$ . Using Lemma 14, we get  $\text{Ess} = \{x_3, x_5\}$  and  $x_4 = 1, x_2 = x_5 + 1, x_1 = x_3 + 1$ , everything mod  $I(P)$ . Thus, let  $\pi(a_1, \dots, a_5) = (a_3, a_5)$ , let  $T = \mathbb{k}[y_1, y_2]$  and let  $\pi^*$  be defined by  $y_1 \mapsto x_3$  and  $y_2 \mapsto x_5$ . We have that  $\prec'$  is lex with  $y_1 \succ' y_2$  and

$$\pi(P) = \{(0, 0), (1, 1), (1, -1), (4, 2)\}.$$

A call to the BM-algorithm with  $\pi(P)$  and  $\prec'$  yields  $B' = \{1, y_2, y_2^2, y_2^3\}$  as the set of monomials outside  $\text{in}(I(\pi(P)))$  and  $\{y_2^4 + 2y_2 - y_2^2 - 2y_2^3, y_1 - y_2^2\}$  as a Grbner basis  $G'$  for  $I(\pi(P))$ . Thus, a Grbner basis  $G$  for  $I(P)$  is

$$\begin{aligned} & \{x_5^4 + 2x_5 - x_5^2 - 2x_5^3, x_3 - x_5^2, \\ & x_4 - \pi^*(\text{Nf}((\pi^*)^{-1}(1), G')), \\ & x_2 - \pi^*(\text{Nf}((\pi^*)^{-1}(x_5), G')) - \pi^*(\text{Nf}((\pi^*)^{-1}(1), G')), \\ & x_1 - \pi^*(\text{Nf}((\pi^*)^{-1}(x_3), G')) - \pi^*(\text{Nf}((\pi^*)^{-1}(1), G'))\} \end{aligned}$$

and the complement of  $\text{in}(I(P))$  is

$$B = \{(\pi^*)^{-1}(1), (\pi^*)^{-1}(y_2), (\pi^*)^{-1}(y_2^2), (\pi^*)^{-1}(y_2^3)\} = \{1, x_5, x_5^2, x_5^3\}.$$

We have  $\pi^*(\text{Nf}((\pi^*)^{-1}(1), G')) = \pi^*(\text{Nf}(1, G')) = \pi^*(1) = 1$  and similarly  $\pi^*(\text{Nf}((\pi^*)^{-1}(x_5), G')) = x_5$  since  $1, x_5 \in B$ . Since  $x_3$  is outside  $B$ , we get  $\pi^*\text{Nf}((\pi^*)^{-1}(x_3), G') = \pi^*(\text{Nf}(y_1, G')) = \pi^*(y_2^2) = x_5^2$ . Thus

$$G = \{x_5^4 + 2x_5 - x_5^2 - 2x_5^3, x_3 - x_5^2, x_4 - 1, x_2 - x_5 - 1, x_1 - x_5^2 - 1\}.$$

Notice that although  $\text{Ess} = \{x_3, x_5\}$  were linearly independent with respect to  $P$ , it did not follow that  $\text{Ess} \subset B$ .

To analyze the complexity of the method above, we first determine the cost of the monomial manipulations.

**Proposition 2.** *Let  $\prec$  be a standard order. An upper bound for the time complexity of the monomial manipulation part of the BM-algorithm using the projection technique is*

$$O(\min(m, n)m^2 \log(m))$$

*Proof.* Suppose that  $m < n$ . We can now use the projection technique described above so that  $\bar{n} \leq m - 1$ . Since the projection technique only affects the arithmetic complexity, we do not need to consider the cost for it in this analysis. By Lemma 5, to determine  $\prec_{\text{Ess}}$  is  $O(1)$ . Everytime we insert an element into  $B$ , we insert at most  $\bar{n}$  elements into  $L$ . Thus, the number of elements in  $L$  is bounded by  $\bar{n}m$ . Thus, the complexity of the monomial manipulation part is dominated by merging a sorted list of  $\bar{n}$  monomials with a sorted list of at most  $\bar{n}m$  monomials, repeated  $m$  times. To compute  $\text{ov}(x_i x^\alpha)$  given  $\text{ov}(x^\alpha)$  is  $O(\log(m))$  by Lemma 3. Thus, each time an element is inserted into  $B$ , it is an  $O(\bar{n} \log(m))$ -operation to create the list of monomials which we will merge with  $L$ . Since we create at most  $m$  such lists, the total time needed for creation is  $O(\bar{n}m \log(m))$ . Using Theorem 1 we see that each merge has time complexity  $O(\bar{n}^2 \max(\log(m/\bar{n}), 1) + \bar{n}m \log(\bar{n} + m))$ .

If  $m \geq n$ , then all arguments hold if we replace  $\bar{n}$  by  $n$ . Thus, in general, each merge has time complexity

$$\begin{aligned} O(\min(m, n)^2 \max(\log(m/\min(m, n)), 1) + \min(m, n)m \log(\min(m, n) + m)) \\ = O(\min(m, n)m \log(m)) \end{aligned}$$

by a straightforward calculation. Since there are exactly  $m - 1$  merges, we get the complexity

$$O(\min(m, n)m^2 \log(m) + \min(m, n)m \log(m)),$$

where the last term comes from the creation process and is negligible.

**Proposition 3.** *Let  $\prec$  be an order defined by an integer matrix  $A$ . Let  $c = \max(|a_{ij}|)$ . We give two upper bounds for the time complexity of the monomial manipulation part of the BM-algorithm using the projection technique, based on two different methods. When  $m \geq n$ , the methods agree and an upper bound is*

$$O(n^2 m \log(cm) + nm^2 \log(n + m) + n^2 \log(c) \log(n)).$$

When  $m < n$ , the first method has the bound

$$O(m^3 \log(cm) + n^2 \log(c) \log(n))$$

to which one needs to add the cost for  $O(nm^2)$  arithmetic operations over  $\mathbb{Q}$ , while the second method has the bound

$$O(nm^2 \log(cm) + n^2 \log(c) \log(n)).$$

*Proof.* First of all we need to determine  $\text{Varord}_{\prec}$ , an  $n^2 \log(c) \log(n)$ -operation by Lemma 8. Suppose that  $m < n$ . We can use the projection technique described above and we now have two choices. Either we use Lemma 9 to construct an  $\bar{n} \times \bar{n}$ -matrix  $A_{\text{Ess}}$  using  $O(nm^2)$  arithmetic operations over  $\mathbb{Q}$ , or we can use the  $n \times \bar{n}$ -submatrix of  $A$ , where we keep the columns that refers to the variables in  $\text{Ess}$ .

In the first case, the cost for computing  $\text{ov}(x_i m)$  given  $\text{ov}(m)$  is  $O(\bar{n} \log(cm))$  by Lemma 6, so the total time needed for the construction of the associated order vectors is  $O(\bar{n}^2 m \log(cm))$ . By Lemma 7 and Theorem 1 we see that each merge has time complexity

$$O(\bar{n}^2 \log(cm) + \bar{n} m \log(\bar{n} + m)) = O(m^2 \log(cm))$$

as we merge a list of  $\bar{n}$  elements with at most  $\bar{n} m$  elements. Since we make  $m - 1$  merges, we deduce that the overall time complexity of the first method is

$$\begin{aligned} O(m^3 \log(cm) + \bar{n}^2 m \log(cm) + n^2 \log(c) \log(n)) \\ = O(m^3 \log(cm) + n^2 \log(c) \log(n)) \end{aligned}$$

to which we need to add  $O(nm^2)$  arithmetic operations over  $\mathbb{Q}$ .

The second method differs from the first in that the vectors are  $n$ -tuples rather than  $\bar{n}$ -tuples. Thus, computing  $\text{ov}(x_i m)$  given  $\text{ov}(m)$  is an  $n \log(cm)$  operation, so the total time needed for the construction process is  $O(n\bar{n}m \log(cm))$ . Each merge requires time  $O(n\bar{n} \log(cm) + \bar{n} m \log(\bar{n} + m)) = O(nm \log(cm))$ , so the overall time complexity of the second method becomes

$$\begin{aligned} O(nm^2 \log(cm) + n\bar{n}m \log(cm) + n^2 \log(c) \log(n)) \\ = O(nm^2 \log(cm) + n^2 \log(c) \log(n)). \end{aligned}$$

When  $m \geq n$ , we do not need to project and the two methods agree. The cost for the construction of the associated order vectors becomes  $O(n^2 m \log(cm))$ , each merge is  $n^2 \log(cm) + nm \log(n + m)$  and thus an upper bound is

$$O(n^2 m \log(cm) + nm^2 \log(n + m) + n^2 m \log(cm) + n^2 \log(c) \log(n)).$$

We are ready to state the main theorem.

**Theorem 2.** *An upper bound for the arithmetic complexity of the BM-algorithm using steps C1, C2', C3, C4 and C5' and the projection technique based on Lemma 14 is*

$$O(nm^2 + \min(m, n)m^3).$$

*To this we need to add the time complexity*

$$O(\min(m, n)m^2 \log(m))$$

*when  $\prec$  is standard.*

*When  $\prec$  is given by a matrix  $A = (a_{ij})$  with  $c = \max(|a_{ij}|)$ , there are two methods to use. When  $m \geq n$ , the methods agree and an upper bound is*

$$O(n^2 m \log(cm) + nm^2 \log(n + m) + n^2 \log(c) \log(n)).$$

*When  $m < n$ , the first method has the bound*

$$O(m^3 \log(cm) + n^2 \log(c) \log(n))$$

*to which one needs to add the cost for  $O(nm^2)$  arithmetic operations over  $\mathbb{Q}$ , while the second method has the bound*

$$O(nm^2 \log(cm) + n^2 \log(c) \log(n)).$$

*Proof.* The complexity for the monomial manipulation part follows from Proposition 2 and Proposition 3.

The call to the BM-algorithm, with or without projection, is  $O(nm^2 + \min(m, n)m^3)$  by Proposition 1. However, if we use the projection technique, we will get the set  $B'$  of monomials outside  $\text{in}(I(\pi(P)))$  and a Grbner basis  $G'$  for  $\text{in}(I(\pi(P)))$ . By Lemma 16,  $B = \pi^*(B')$  and by Lemma 17,

$$G = \pi^*(G') \sqcup \{x_k - \sum_j c_{kj} \pi^*(\text{Nf}((\pi^*)^{-1}(x_{i_j}), G'))\}.$$

So we are done if we can show that each  $x_k - \sum_j c_{kj} \pi^*(\text{Nf}((\pi^*)^{-1}(x_{i_j}), G'))$  is computable within  $O(m^2)$  arithmetic operations. To get a short proof, we will not use the information  $x_k - \sum_j c_{kj} x_{i_j} \in I$ . Instead we compute the evaluation vector  $(x_k(p_1), \dots, x_k(p_m)) = (p_{1k}, \dots, p_{mk})$  and write it as a linear combination of the elements in  $B$ , an operation which requires  $O(m^2)$  arithmetic operations. Since  $B$  is a basis, the linear combination will then equal  $\sum_j c_{kj} \pi^*(\text{Nf}((\pi^*)^{-1}(x_{i_j}), G'))$ .

The following corollary states that our version of the BM-algorithm is preferable to the EssGB-algorithm [7].

**Corollary 1.** *When  $m < n$  and the order is standard, the BM-algorithm using steps C1, C2', C3, C4 and C5' and the projection technique based on Lemma 14 has arithmetic complexity*

$$O(nm^2 + m^4).$$

*To this we need to add the time complexity*

$$O(m^3 \log(m)).$$

### 3.3 Applications to the FGLM-Algorithm and for Ideals Defined by Functionals

As was noticed in [10], both the FGLM- and the BM-algorithm are instances of definitions of an ideal defined by means of a finite set of functionals  $L_i : \mathbb{k}[x_1, \dots, x_n] \rightarrow \mathbb{k}$ , such that  $I$  is in the kernel of  $\Psi : \mathbb{k}[x_1, \dots, x_n] \rightarrow \mathbb{k}^m, \Psi(f) = (L_1(f), \dots, L_m(f))$ . For the BM-setting, the functionals are defined by  $L_i(f) = f(p_i)$  and in the FGLM-setting, the functionals are defined by  $\text{Nf}(f, G_1) = \sum L_i(f)e_i$ . In [10], a list of different problems that can be seen as instances of an ideal defined by functionals is given.

If we use the steps C1,C2',F3,C4,C5' of the BM-algorithm where F3 is defined below, we obtain Algorithm 1 in [10].

**F3** Compute  $\Psi(t) = (b_1, \dots, b_m)$  and reduce it against the rows of  $C$  to obtain

$$(v_1, \dots, v_m) = (b_1, \dots, b_m) - \sum_i a_i(c_{i1}, \dots, c_{im}) \quad a_i \in \mathbb{k}.$$

It is reported in [10] Theorem 5.1, that Algorithm 1 in [10] needs  $O(nm^3 + fnm^2)$  arithmetic operations, where  $f$  denotes the cost of evaluating a functional. However, as for the BM-algorithm, one can replace the term  $nm^3$  by  $m^4 + nm^2$ . Since we have shown (Proposition 1) that the number of calls to C3 equals  $|G| + m = n + \min(m, n)m + m$ , one can replace the term  $fnm^2$  by  $fnm + f \min(m, n)m^2 + fm$ . Thus, Algorithm 1 in [10] uses  $O(\min(m, n)m^3 + nm^2 + fnm + f \min(m, n)m^2)$  arithmetic operations. In the BM- or the FGLM-situation, it is shown in [10] that  $f = 1$ , by a recursive argument.

To the arithmetic complexity one needs to add the cost for the monomial manipulations, which is the same as for the BM-algorithm, since it is clear that we can use the projection technique described in Lemma 14 if we replace

$$x_{i_{n-j+1}}(P) \notin \text{span}_{\mathbb{k}}\{1(P), E_{j-1}(P)\}$$

by

$$\Psi(x_{i_{n-j}}) \notin \text{span}_{\mathbb{k}}\{\Psi(1), \Psi(x_{i_{n-j+2}}), \dots, \Psi(x_{i_n})\}.$$

It follows that Theorem 2 can be lifted to the general setting of ideals defined by functionals. We state

**Theorem 3.** *An upper bound for the arithmetic complexity of Algorithm 1 in [10] using the projection technique is*

$$O(\min(m, n)m^3 + nm^2 + fnm + f \min(m, n)m^2).$$

*To this we need to add the time complexity*

$$O(\min(m, n)m^2 \log(m))$$

*when  $\prec$  is standard.*

When  $\prec$  is given by a matrix  $A = (a_{ij})$  with  $c = \max(|a_{ij}|)$ , there are two methods to use. When  $m \geq n$ , the methods agree and an upper bound is

$$O(n^2 m \log(cm) + nm^2 \log(n+m) + n^2 \log(c) \log(n)).$$

When  $m < n$ , the first method has the bound

$$O(m^3 \log(cm) + n^2 \log(c) \log(n))$$

to which one needs to add the cost for  $O(nm^2)$  arithmetic operations over  $\mathbb{Q}$ , while the second method has the bound

$$O(nm^2 \log(cm) + n^2 \log(c) \log(n)).$$

## 4 Discussion and Future Work

The projection idea can be used for ideals defined by projective points and also in the non-commutative versions of the FGLM-algorithm, given in [2]. It should be remarked that the monomial manipulations with respect to non-commuting variables is computationally harder than those for commuting variables.

Future work involves the question: Why do we need a Grbner basis for ideals defined by vanishing points? In the biological applications, where one is primarily interested in normal form computations, it seems enough to compute a set  $B$  such that  $[B]$  is a  $\mathbb{k}$ -basis for  $S/I$ .

## References

1. Abbott, J., Bigatti, A., Kreuzer, M., Robbiano, L.: Computing ideals of points. *J. Symb. Comput.* 30(4), 341–356 (2000)
2. Borges-Trenard, M.A., Borges-Quintana, M., Mora, T.: Computing Gröbner bases by FGLM techniques in a non-commutative setting. *J. Symb. Comput.* 30(4), 429–449 (2000)
3. Buchberger, B., Möller, M.: The construction of multivariate polynomials with preassigned zeroes. In: Calmet, J. (ed.) *ISSAC 1982 and EUROCAM 1982*. LNCS, vol. 144, pp. 24–31. Springer, Heidelberg (1982)
4. Faugre, J.C., Gianni, P., Lazard, D., Mora, T.: Efficient Computation of Zero-Dimensional Gröbner Basis by Change of Ordering. *J. Symb. Comput.* 16(4), 329–344 (1993)
5. Frer, M.: Faster integer multiplication. In: *Proceedings of the 39th ACM STOC 2007 conference*, pp. 57–66 (2007)
6. Just, W., Stigler, B.: Computing Gröbner bases of ideals of few points in high dimensions. *Communications in Computer Algebra* 40(3), 65–96 (2006)
7. Just, W., Stigler, B.: Efficiently computing Groebner bases of ideals of points (2007) (arXiv:0711.3475)
8. Kühnle, K., Mayr, E.W.: Exponential space computation of Gröbner bases. In: *ISSAC 1996: Proceedings of the 1996 international symposium on Symbolic and algebraic computation*, pp. 63–71 (1996)

9. Laubenbacher, R., Stigler, B.: A computational algebra approach to the reverse-engineering of gene regulatory networks. *J. Theor. Biol.* 229, 523–537 (2004)
10. Marinari, M., Möller, H.M., Mora, T.: Gröbner bases of ideals defined by functionals with an application to ideals of projective points. *Applicable Algebra in Engineering, Communication and Computing* 4, 103–145 (1993)
11. Robbiano, L.: Term orderings on the polynomial ring. In: Caviness, B.F. (ed.) *ISSAC 1985 and EUROCAL 1985*. LNCS, vol. 204, pp. 513–517. Springer, Heidelberg (1985)

# Invited Talk: Decoding Cyclic Codes: The Cooper Philosophy (Extended Abstract)

Teo Mora<sup>1</sup> and Emmanuela Orsini<sup>2</sup>

<sup>1</sup> Department of Mathematics,  
University of Genoa, Italy  
theomora@disi.unige.it

<sup>2</sup> Department of Mathematics,  
University of Milan, Italy  
orsini@posso.dm.unipi.it

In 1990, Cooper [9, 10] suggested to use Gröbner basis [3, 4] computation in order to deduce error locator polynomials of cyclic codes.

Following his idea, Chen et al. [6, 7, 8] suggested a general algorithm to pursue Cooper's approach. The aim of the talk is to follow, on an illuminating example, the arguments which, through a series of papers [5, 12, 13], led to the following result (for an extension see [11]):

**Theorem 1.** [13] *For each  $[n, k, d]$  binary cyclic code  $C$  with  $n$  odd, denoting  $\mathbb{F}$  the splitting field of  $x^n - 1$  over  $\mathbb{Z}_2$ , a proper Gröbner basis computation allows to produce a polynomial  $\mathcal{L} \in \mathbb{Z}_2[X, z]$ , where  $X = (x_1, \dots, x_{n-k})$  which satisfies the following properties:*

1.  $\mathcal{L}(X, z) = z^t + a_{t-1}(X)z^{t-1} + \dots + a_0(X)$ , with  $a_j \in \mathbb{Z}_2[X]$ ,  $0 \leq j \leq t-1$ ;
2. *given a syndrome vector  $\mathbf{s} = (s_1, \dots, s_{n-k}) \in (\mathbb{F})^{n-k}$  corresponding to an error with weight  $\mu \leq t$ , if we evaluate the  $X$  variables in  $\mathbf{s}$ , then the  $t$  roots of  $\mathcal{L}(\mathbf{s}, z)$  are the  $\mu$  error locations plus zero counted with multiplicity  $t - \mu$ .*

We illustrate the efficiency of this approach on the recent results discussed in [14, 15] and we also discuss an alternative approach to the solution of the Cooper problem proposed in [1, 2].

## References

1. Augot, D., Bardet, M., Faugere, J.C.: Efficient decoding of (binary) cyclic codes above the correction capacity of the code using Gröbner bases. In: Proc. IEEE Int. Symp. Information Theory 2003 (2003)
2. Augot, D., Bardet, M., Faugere, J.C.: On formulas for decoding binary cyclic codes. In: Proc. IEEE Int. Symp. Information Theory 2007 (2007)
3. Buchberger, B.: Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal, Ph. D. Thesis, Innsbruck (1965)

4. Buchberger, B.: Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems. *Aeq. Math.* 4, 374–383 (1970)
5. Caboara, M.: The Chen-Reed-Helleseth-Truong Decoding Algorithm and the Gianni-Kalkbrenner Gröbner Shape Theorem. *J. AAECC* 13 (2002)
6. Chen, X., Reed, I.S., Helleseth, T., Truong, K.: Use of Gröbner Bases to Decode Binary Cyclic Codes up to the True Minimum Distance. *IEEE Trans. on Inf. Th.* 40, 1654–1661 (1994)
7. Chen, X., Reed, I.S., Helleseth, T., Truong, K.: General Principles for the Algebraic Decoding of Cyclic Codes. *IEEE Trans. on Inf. Th.* 40, 1661–1663 (1994)
8. Chen, X., Reed, I.S., Helleseth, T., Truong, K.: Algebraic decoding of cyclic codes: A polynomial Ideal Point of View. *Contemporary Mathematics* 168, 15–22 (1994)
9. Cooper III, A.B.: Direct solution of BCH decoding equations. In: Arıkan, E. (ed.) *Communication, Control and Signal Processing*, pp. 281–286. Elsevier, Amsterdam (1990)
10. Cooper III, A.B.: Finding BCH error locator polynomials in one step. *Electronic Letters* 27, 2090–2091 (1991)
11. Giorgetti, M., Sala, M.: A commutative algebra approach to linear codes, BCRI preprint, 58, UCC Cork, Ireland (2006), [www.bcricork.ie](http://www.bcricork.ie)
12. Loustau, P., York, E.V.: On the decoding of cyclic codes using Gröbner bases. *J. AAECC* 8, 469–483 (1997)
13. Orsini, E., Sala, M.: Correcting errors and erasures via the syndrome variety. *J. Pure Appl. Algebra* 200, 191–226 (2005)
14. Orsini, E., Sala, M.: General error locator polynomials for binary cyclic codes with  $t \leq 2$  and  $n < 63$ . *IEEE Trans. Inform. Theory* 53, 1095–1107 (2007)
15. Orsini, E., Sala, M.: General error locator polynomials for binary cyclic codes with  $t \leq 2$  and  $n < 63$ , BCRI preprint (2005), <http://www.bcricork.ie>

# Kernel Dimension for Some Families of Quaternary Reed-Muller Codes<sup>\*</sup>

J. Pernas, J. Pujol, and M. Villanueva

Dept. of Information and Communications Engineering,  
Universitat Autònoma de Barcelona, Spain  
{jaume.pernas, jaume.pujol, merce.villanueva}@autonoma.edu

**Abstract.** Recently, new families of quaternary linear Reed-Muller codes such that, after the Gray map, the corresponding  $\mathbb{Z}_4$ -linear codes have the same parameters and properties as the codes in the usual binary linear Reed-Muller family have been introduced. A structural invariant, the kernel dimension, for binary codes can be used to classify these  $\mathbb{Z}_4$ -linear codes. The kernel dimension for these  $\mathbb{Z}_4$ -linear codes is established generalizing the known results about the kernel dimension for  $\mathbb{Z}_4$ -linear Hadamard and  $\mathbb{Z}_4$ -linear extended 1-perfect codes.

**Keywords:** Kernel dimension, quaternary codes, Reed-Muller codes,  $\mathbb{Z}_4$ -linear codes.

## 1 Introduction

Let  $\mathbb{Z}_2$  and  $\mathbb{Z}_4$  be the ring of integers modulo 2 and modulo 4, respectively. Let  $\mathbb{Z}_2^n$  be the set of all binary vectors of length  $n$  and let  $\mathbb{Z}_4^n$  be the set of all quaternary vectors of length  $n$ . Any nonempty subset  $C$  of  $\mathbb{Z}_2^n$  is a binary code and a subgroup of  $\mathbb{Z}_2^n$  is called a *binary linear code* or a  $\mathbb{Z}_2$ -linear code. Equivalently, any nonempty subset  $C$  of  $\mathbb{Z}_4^n$  is a quaternary code and a subgroup of  $\mathbb{Z}_4^n$  is called a *quaternary linear code*.

The *Hamming distance*  $d_H(u, v)$  between two vectors  $u, v \in \mathbb{Z}_2^n$  is the number of coordinates in which  $u$  and  $v$  differ. The *Hamming weight* of a vector  $u \in \mathbb{Z}_2^n$ , denoted by  $w_H(u)$ , is the number of nonzero coordinates of  $u$ . The *minimum Hamming distance* of a binary code  $C$  is the minimum value of  $d_H(u, v)$  for  $u, v \in C$  satisfying  $u \neq v$ . The *minimum Hamming weight* of a binary code  $C$ , denoted by  $w_{min}(C)$ , is the minimum value of  $w_H(u)$ , for  $u \in C \setminus \{0\}$ .

We define the *Lee weights* over the elements in  $\mathbb{Z}_4$  as:  $w_L(0) = 0$ ,  $w_L(1) = w_L(3) = 1$ ,  $w_L(2) = 2$ . The *Lee weight* of a vector  $u \in \mathbb{Z}_4^n$ , denoted by  $w_L(u)$ , is the addition of the weights of its coordinates, whereas the *Lee distance*  $d_L(u, v)$  between two vectors  $u, v \in \mathbb{Z}_4^n$  is  $d_L(u, v) = w_L(u - v)$ . The *minimum Lee distance* of a quaternary code  $C$  is the minimum value of  $d_L(u, v)$  for  $u, v \in C$  satisfying  $u \neq v$ . The *minimum Lee weight* of a quaternary code  $C$ , denoted by  $w_{min}(C)$ , is the minimum value of  $w_L(0, u)$ , for  $u \in C \setminus \{0\}$ .

---

<sup>\*</sup> This work was supported in part by the Spanish MEC and the European FEDER under Grant MTM2006-03250.

The Gray map,  $\phi : \mathbb{Z}_4^n \longrightarrow \mathbb{Z}_2^{2n}$  given by  $\phi(v_1, \dots, v_n) = (\varphi(v_1), \dots, \varphi(v_n))$  where  $\varphi(0) = (0, 0)$ ,  $\varphi(1) = (0, 1)$ ,  $\varphi(2) = (1, 1)$ ,  $\varphi(3) = (1, 0)$ , is an isometry which transforms Lee distances defined in a quaternary code  $\mathcal{C}$  over  $\mathbb{Z}_4^n$  to Hamming distances defined in the corresponding binary code  $C = \phi(\mathcal{C})$ . Therefore,  $w_{min}(\mathcal{C}) = w_{min}(C)$ . Note that the binary length of the binary code  $C$  is  $N = 2n$ .

Let  $\mathcal{C}$  be a quaternary linear code. Since  $\mathcal{C}$  is a subgroup of  $\mathbb{Z}_4^n$ , it is isomorphic to an abelian structure  $\mathbb{Z}_2^\gamma \times \mathbb{Z}_4^\delta$ . Therefore,  $\mathcal{C}$  is of type  $2^\gamma 4^\delta$  as a group, it has  $|\mathcal{C}| = 2^{\gamma+2\delta}$  codewords and  $2^{\gamma+\delta}$  codewords of order two. Moreover, the binary image  $C = \phi(\mathcal{C})$  of any quaternary linear code  $\mathcal{C}$  of length  $n$  and type  $2^\gamma 4^\delta$  is called a  $\mathbb{Z}_4$ -linear code of binary length  $N = 2n$  and type  $2^\gamma 4^\delta$ .

Two binary codes  $C_1$  and  $C_2$  of length  $n$  are said to be *isomorphic* if there is a coordinate permutation  $\pi$  such that  $C_2 = \{\pi(c) : c \in C_1\}$ . They are said to be *equivalent* if there is a vector  $a \in \mathbb{Z}_2^n$  and a coordinate permutation  $\pi$  such that  $C_2 = \{a + \pi(c) : c \in C_1\}$  [14]. Two quaternary linear codes  $\mathcal{C}_1$  and  $\mathcal{C}_2$  both of length  $n$  and type  $2^\gamma 4^\delta$  are said to be *monomially equivalent*, if one can be obtained from the other by permuting the coordinates and (if necessary) changing the signs of certain coordinates. They are said to be *permutation equivalent* if they differ only by a permutation of coordinates [12]. Note that if two quaternary linear codes  $\mathcal{C}_1$  and  $\mathcal{C}_2$  are monomially equivalent, then the corresponding  $\mathbb{Z}_4$ -linear codes  $C_1 = \phi(\mathcal{C}_1)$  and  $C_2 = \phi(\mathcal{C}_2)$  are isomorphic.

Two structural invariants for binary codes are the rank and dimension of the kernel. The *rank* of a binary code  $C$ , denoted by  $r_C$ , is simply the dimension of  $\langle C \rangle$ , which is the linear span of the codewords of  $C$ . The *kernel* of a binary code  $C$ , denoted by  $K(C)$ , is the set of vectors that leave  $C$  invariant under translation, i.e.  $K(C) = \{x \in \mathbb{Z}_2^n : C + x = C\}$ . If  $C$  contains the all-zero vector, then  $K(C)$  is a binary linear subcode of  $C$ . In general,  $C$  can be written as the union of cosets of  $K(C)$ , and  $K(C)$  is the largest such linear code for which this is true [1]. The dimension of the kernel of  $C$  will be denoted by  $k_C$ .

These two invariants do not give a full classification of binary codes, since two nonequivalent binary codes could have the same rank and dimension of the kernel. In spite of that, they can help in classification, since if two binary codes have different ranks or dimensions of the kernel, they are nonequivalent.

In [10], Hammons et al. showed that several families of binary codes are  $\mathbb{Z}_4$ -linear. In particular, they considered the binary linear Reed-Muller family of codes, denoted by  $RM$ , and proved that the binary linear  $r$ th-order Reed-Muller code  $RM(r, m)$  of length  $2^m$  is  $\mathbb{Z}_4$ -linear for  $r = 0, 1, 2, m - 1, m$  and is not  $\mathbb{Z}_4$ -linear for  $r = m - 2$  ( $m \geq 5$ ). In a subsequent work [11], Hou et al. proved that  $RM(r, m)$  is not  $\mathbb{Z}_4$ -linear for  $3 \leq r \leq m - 2$  ( $m \geq 5$ ).

It is well-known that an easy way to build the binary linear Reed-Muller family of codes  $RM$  is using the Plotkin construction [14]. In [16],[17], new quaternary Plotkin constructions were introduced to build new families of quaternary linear Reed-Muller codes, denoted by  $\mathcal{RM}_s$ . The quaternary linear Reed-Muller codes  $\mathcal{RM}_s(r, m)$  of length  $2^{m-1}$ , for  $m \geq 1$ ,  $0 \leq r \leq m$  and  $0 \leq s \leq \lfloor \frac{m-1}{2} \rfloor$ , in these new families satisfy that the corresponding  $\mathbb{Z}_4$ -linear codes have the same

parameters and properties (length, dimension, minimum distance, inclusion and duality relationship) as the binary linear codes in the well-known  $RM$  family. Contrary to the binary linear case, where there is only one family, in the quaternary case there are  $\lfloor \frac{m+1}{2} \rfloor$  families for each value of  $m$ . These families will be distinguished using subindexes  $s$  from the set  $\{0, \dots, \lfloor \frac{m-1}{2} \rfloor\}$ .

The dimension of the kernel and rank have been studied for some families of  $\mathbb{Z}_4$ -linear codes [2],[5],[6],[13],[15]. In the  $RM$  family, the  $RM(1, m)$  and  $RM(m-2, m)$  binary codes are a linear Hadamard and extended 1-perfect code, respectively. Recall that a Hadamard code of length  $n = 2^m$  is a binary code with  $2n$  codewords and minimum Hamming distance  $n/2$ , and an extended 1-perfect code of length  $n = 2^m$  is a binary code with  $2^{n-m}$  codewords and minimum Hamming distance 4. Equivalently, in the  $\mathcal{RM}_s$  families, the corresponding  $\mathbb{Z}_4$ -linear code of any  $\mathcal{RM}_s(1, m)$  and  $\mathcal{RM}_s(m-2, m)$  is a Hadamard and extended 1-perfect code, respectively [16],[17]. For the corresponding  $\mathbb{Z}_4$ -linear codes of  $\mathcal{RM}_s(1, m)$  and  $\mathcal{RM}_s(m-2, m)$ , the rank and kernel dimension were studied and computed in [6],[13],[15]. Specifically,

$$k_H = \begin{cases} \gamma + \delta + 1 & \text{if } s \geq 2 \\ \gamma + 2\delta & \text{if } s = 0, 1 \end{cases} \quad \text{and} \quad k_P = \begin{cases} \bar{\gamma} + \bar{\delta} + 1 & \text{if } s \geq 2 \\ \bar{\gamma} + \bar{\delta} + 2 & \text{if } s = 1 \\ \bar{\gamma} + \bar{\delta} + m & \text{if } s = 0. \end{cases} \quad (1)$$

where  $H = \phi(\mathcal{RM}_s(1, m))$  of type  $2^\gamma 4^\delta$  and  $P = \phi(\mathcal{RM}_s(m-2, m))$  of type  $2^{\bar{\gamma}} 4^{\bar{\delta}}$ .

The aim of this paper is the study of the dimension of the kernel for the quaternary linear Reed-Muller families of codes  $\mathcal{RM}_s$ , generalizing the known results about the kernel dimension for the  $\mathcal{RM}_s(1, m)$  and  $\mathcal{RM}_s(m-2, m)$  codes. The paper is organized as follows. In Section 2, we recall some properties related to quaternary linear codes and the kernel of these codes. Moreover, we describe the construction of the  $\mathcal{RM}_s$  families of codes. In Section 3, we establish the dimension of the kernel for all codes in the  $\mathcal{RM}_s$  family with  $s = 0$ . In Section 4, we give the main results about the kernel dimension for all the  $\mathcal{RM}_s$  families. Finally, the conclusions are given in Section 5.

MAGMA is a software package designed to solve computationally hard problems in algebra, number theory, geometry and combinatorics. Currently it supports the basic facilities for linear codes over integer residue rings and Galois rings (see [7]), including additional functionality for the special case of codes over  $\mathbb{Z}_4$ , or equivalently quaternary linear codes. New functions that expand the current functionality for codes over  $\mathbb{Z}_4$  have been developed by the authors as a new package. Specifically, these functions allow to construct the  $\mathcal{RM}_s$  families and some Plotkin constructions for quaternary linear codes. Moreover, efficient functions for computing the rank and dimension of the kernel of any quaternary linear code are included. A beta version of this new package and the manual with the description of all functions can be downloaded from the web page <http://www.ccg.uab.cat>.

## 2 Preliminaries

### 2.1 Quaternary Linear Codes

Let  $\mathcal{C}$  be a quaternary linear code of length  $n$  and type  $2^\gamma 4^\delta$ . Although  $\mathcal{C}$  is not a free module, every codeword is uniquely expressible in the form

$$\sum_{i=1}^{\gamma} \lambda_i u_i + \sum_{j=1}^{\delta} \mu_j v_j,$$

where  $\lambda_i \in \mathbb{Z}_2$  for  $1 \leq i \leq \gamma$ ,  $\mu_j \in \mathbb{Z}_4$  for  $1 \leq j \leq \delta$  and  $u_i, v_j$  are vectors in  $\mathbb{Z}_4^n$  of order two and four, respectively. The vectors  $u_i, v_j$  give us a generator matrix  $\mathcal{G}$  of size  $(\gamma + \delta) \times n$  for the code  $\mathcal{C}$ . Moreover,  $\mathcal{G}$  will also be used to denote the set of all vectors  $u_i, v_j$ .

In [10], it was shown that any quaternary linear code of type  $2^\gamma 4^\delta$  is permutation equivalent to a quaternary linear code with a canonical generator matrix of the form

$$\begin{pmatrix} 2T & 2I_\gamma & \mathbf{0} \\ S & R & I_\delta \end{pmatrix},$$

where  $R, T$  are matrices over  $\mathbb{Z}_2$  of size  $\delta \times \gamma$  and  $\gamma \times (n - \gamma - \delta)$ , respectively; and  $S$  is a matrix over  $\mathbb{Z}_4$  of size  $\delta \times (n - \gamma - \delta)$ .

The concepts of duality for quaternary linear codes were also studied in [10], where the inner product for any two vectors  $u, v \in \mathbb{Z}_4^n$  is defined as

$$u \cdot v = \sum_{i=1}^n u_i v_i \in \mathbb{Z}_4.$$

Then, the *dual code* of  $\mathcal{C}$ , denoted by  $\mathcal{C}^\perp$ , is defined in the standard way

$$\mathcal{C}^\perp = \{v \in \mathbb{Z}_4^n : u \cdot v = 0 \text{ for all } u \in \mathcal{C}\}.$$

The corresponding binary code  $\phi(\mathcal{C}^\perp)$  is denoted by  $C_\perp$  and called the  $\mathbb{Z}_4$ -*dual code* of  $\mathcal{C}$ . Moreover, the dual code  $\mathcal{C}^\perp$ , which is also a quaternary linear code, is of type  $2^\gamma 4^{n-\gamma-\delta}$ .

The all-zero and all-one vector will be denoted by  $\mathbf{0}$  and  $\mathbf{1}$ , respectively. It will be clear by the context whether we refer to binary vectors or quaternary vectors.

Let  $\mathcal{C}$  be a quaternary linear code and let  $C = \phi(\mathcal{C})$  be the corresponding  $\mathbb{Z}_4$ -linear code with kernel  $K(C)$ . The kernel of  $\mathcal{C}$ , denoted by  $\mathcal{K}(\mathcal{C})$ , is defined as the inverse Gray map image of  $K(C)$ , that is  $\mathcal{K}(\mathcal{C}) = \phi^{-1}(K(C))$ . Furthermore, the dimension of the kernel of  $\mathcal{C}$  is defined as the dimension of the kernel of  $C = \phi(\mathcal{C})$ , and also denoted by  $k_{\mathcal{C}}$ .

Let  $u * v$  denote the component-wise product for any  $u, v \in \mathbb{Z}_4^n$ .

**Lemma 1** ([8],[9]). *Let  $\mathcal{C}$  be a quaternary linear code. Then,*

$$\mathcal{K}(\mathcal{C}) = \{u : u \in \mathcal{C} \text{ and } 2u * v \in \mathcal{C}, \forall v \in \mathcal{C}\}.$$

Note that if  $\mathcal{G}$  is a generator matrix of a quaternary linear code  $\mathcal{C}$ , then  $u \in \mathcal{K}(\mathcal{C})$  if and only if  $u \in \mathcal{C}$  and  $2u * v \in \mathcal{C}$  for all  $v \in \mathcal{G}$ . Moreover, by Lemma 1, all codewords of order two in  $\mathcal{C}$  belong to  $\mathcal{K}(\mathcal{C})$ . It is also clear that if the vector  $\mathbf{1}$  belongs to  $\mathcal{C}$ , then it is also in  $\mathcal{K}(\mathcal{C})$ . Finally, note that  $\mathcal{K}(\mathcal{C})$  is a linear subcode of  $\mathcal{C}$  [8],[9].

## 2.2 Quaternary Linear Reed-Muller Codes

Recall that a binary linear  $r$ th-order Reed-Muller code  $RM(r, m)$  with  $0 \leq r \leq m$  and  $m \geq 2$  can be described using the Plotkin construction as follows [14]:

$$RM(r, m) = \{(u|u + v) : u \in RM(r, m - 1), v \in RM(r - 1, m - 1)\},$$

where  $RM(0, m)$  is the repetition code  $\{0, \mathbf{1}\}$ ,  $RM(m, m)$  is the universe code, and “|” denotes concatenation. For  $m = 1$ , there are only two codes: the repetition code  $RM(0, 1)$  and the universe code  $RM(1, 1)$ . This  $RM$  family has the parameters and properties quoted in the following proposition.

**Proposition 2 ([14]).** *A binary linear  $r$ th-order Reed-Muller code  $RM(r, m)$  with  $m \geq 1$  and  $0 \leq r \leq m$  has the following parameters and properties:*

1. the length is  $n = 2^m$ ;
2. the minimum Hamming distance is  $d = 2^{m-r}$ ;
3. the dimension is  $k = \sum_{i=0}^r \binom{m}{i}$ ;
4. the code  $RM(r - 1, m)$  is a subcode of  $RM(r, m)$  for  $0 < r \leq m$ ;
5. the code  $RM(r, m)$  is the dual code of  $RM(m - 1 - r, m)$  for  $0 \leq r < m$ .

In the recent literature [2],[3],[10],[18],[19] several families of quaternary linear codes have been proposed and studied trying to generalize the  $RM$  family. However, when the corresponding  $\mathbb{Z}_4$ -linear codes are taken, they do not satisfy all the properties quoted in Proposition 2. In [16],[17], new quaternary linear Reed-Muller families,  $\mathcal{RM}_s$ , such that the corresponding  $\mathbb{Z}_4$ -linear codes have the parameters and properties described in Proposition 2, were proposed. The following two Plotkin constructions are necessary to generate these new  $\mathcal{RM}_s$  families.

**Definition 3 (Plotkin Construction).** *Let  $\mathcal{A}$  and  $\mathcal{B}$  be two quaternary linear codes of length  $n$ , types  $2^{\gamma_A} 4^{\delta_A}$  and  $2^{\gamma_B} 4^{\delta_B}$ , and minimum distances  $d_A$  and  $d_B$ , respectively. A new quaternary linear code  $\mathcal{PC}(\mathcal{A}, \mathcal{B})$  is defined as*

$$\mathcal{PC}(\mathcal{A}, \mathcal{B}) = \{(u|u + v) : u \in \mathcal{A}, v \in \mathcal{B}\}.$$

It is easy to see that if  $\mathcal{G}_A$  and  $\mathcal{G}_B$  are generator matrices of  $\mathcal{A}$  and  $\mathcal{B}$ , respectively, then the matrix

$$\mathcal{G}_{PC} = \begin{pmatrix} \mathcal{G}_A & \mathcal{G}_A \\ 0 & \mathcal{G}_B \end{pmatrix}$$

is a generator matrix of the code  $\mathcal{PC}(\mathcal{A}, \mathcal{B})$ . Moreover, the code  $\mathcal{PC}(\mathcal{A}, \mathcal{B})$  is of length  $2n$ , type  $2^{\gamma_A + \gamma_B} 4^{\delta_A + \delta_B}$ , and minimum distance  $d = \min\{2d_A, d_B\}$  [16],[17].

**Definition 4 (BQ-Plotkin Construction).** Let  $\mathcal{A}$ ,  $\mathcal{B}$ , and  $\mathcal{C}$  be three quaternary linear codes of length  $n$ ; types  $2^{\gamma_A}4^{\delta_A}$ ,  $2^{\gamma_B}4^{\delta_B}$ , and  $2^{\gamma_C}4^{\delta_C}$ ; and minimum distances  $d_A$ ,  $d_B$ , and  $d_C$ , respectively. Let  $\mathcal{G}_A$ ,  $\mathcal{G}_B$ , and  $\mathcal{G}_C$  be generator matrices of the codes  $\mathcal{A}$ ,  $\mathcal{B}$ , and  $\mathcal{C}$ , respectively. A new code  $\mathcal{BQ}(\mathcal{A}, \mathcal{B}, \mathcal{C})$  is defined as the quaternary linear code generated by

$$\mathcal{G}_{BQ} = \begin{pmatrix} \mathcal{G}_A & \mathcal{G}_A & \mathcal{G}_A & \mathcal{G}_A \\ 0 & \mathcal{G}'_B & 2\mathcal{G}'_B & 3\mathcal{G}'_B \\ 0 & 0 & \hat{\mathcal{G}}_B & \hat{\mathcal{G}}_B \\ 0 & 0 & 0 & \mathcal{G}_C \end{pmatrix},$$

where  $\mathcal{G}'_B$  is the matrix obtained from  $\mathcal{G}_B$  after switching twos by ones in their  $\gamma_B$  rows of order two, and  $\hat{\mathcal{G}}_B$  is the matrix obtained from  $\mathcal{G}_B$  after removing their  $\gamma_B$  rows of order two.

The code  $\mathcal{BQ}(\mathcal{A}, \mathcal{B}, \mathcal{C})$  is of length  $4n$ , type  $2^{\gamma_A+\gamma_C}4^{\delta_A+\gamma_B+2\delta_B+\delta_C}$ , and minimum distance  $d = \min\{4d_A, 2d_B, d_C\}$  [16],[17].

Now, the quaternary linear Reed-Muller codes  $\mathcal{RM}_s(r, m)$  of length  $2^{m-1}$ , for  $m \geq 1$ ,  $0 \leq r \leq m$ , and  $0 \leq s \leq \lfloor \frac{m-1}{2} \rfloor$ , will be defined. For the recursive construction it will be convenient to define them also for  $r < 0$  and  $r > m$ . We begin by considering the trivial cases. The code  $\mathcal{RM}_s(r, m)$  with  $r < 0$  is defined as the zero code. The code  $\mathcal{RM}_s(0, m)$  is defined as the repetition code with only the all-zero and all-two vectors. The code  $\mathcal{RM}_s(r, m)$  with  $r \geq m$  is defined as the whole space  $\mathbb{Z}_4^{m-1}$ . For  $m = 1$ , there is only one family with  $s = 0$ , and in this family there are only the zero, repetition and universe codes for  $r < 0$ ,  $r = 0$  and  $r \geq 1$ , respectively. In this case, the generator matrix of  $\mathcal{RM}_0(0, 1)$  is  $\mathcal{G}_{0(0,1)} = (2)$  and the generator matrix of  $\mathcal{RM}_0(1, 1)$  is  $\mathcal{G}_{0(1,1)} = (1)$ .

For any  $m \geq 2$ , given  $\mathcal{RM}_s(r, m-1)$  and  $\mathcal{RM}_s(r-1, m-1)$  codes, where  $0 \leq s \leq \lfloor \frac{m-2}{2} \rfloor$ , the  $\mathcal{RM}_s(r, m)$  code can be constructed in a recursive way using the Plotkin construction given by Definition 3 as follows:

$$\mathcal{RM}_s(r, m) = \mathcal{PC}(\mathcal{RM}_s(r, m-1), \mathcal{RM}_s(r-1, m-1)).$$

For example, for  $m = 2$ , the generator matrices of  $\mathcal{RM}_0(r, 2)$ ,  $0 \leq r \leq 2$ , are the following:

$$\mathcal{G}_{0(0,2)} = (2 \ 2); \quad \mathcal{G}_{0(1,2)} = \begin{pmatrix} 0 & 2 \\ 1 & 1 \end{pmatrix}; \quad \mathcal{G}_{0(2,2)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Note that when  $m$  is odd, the  $\mathcal{RM}_s$  family with  $s = \frac{m-1}{2}$  can not be generated using the Plotkin construction. In this case, for any  $m \geq 3$ ,  $m$  odd and  $s = \frac{m-1}{2}$ , given  $\mathcal{RM}_{s-1}(r, m-2)$ ,  $\mathcal{RM}_{s-1}(r-1, m-2)$  and  $\mathcal{RM}_{s-1}(r-2, m-2)$ , the  $\mathcal{RM}_s(r, m)$  code can be constructed using the BQ-Plotkin construction given by Definition 4 as follows:

$$\mathcal{RM}_s(r, m) = \mathcal{BQ}(\mathcal{RM}_{s-1}(r, m-2), \mathcal{RM}_{s-1}(r-1, m-2), \mathcal{RM}_{s-1}(r-2, m-2)).$$

For example, for  $m = 3$ , there are two families. The  $\mathcal{RM}_0$  family can be generated using the Plotkin construction. On the other hand, the  $\mathcal{RM}_1$  family has

to be generated using the BQ-Plotkin construction. The generator matrices of  $\mathcal{RM}_1(r, 3)$ ,  $0 \leq r \leq 3$ , are the following:  $\mathcal{G}_{1(0,3)} = (2\ 2\ 2\ 2)$ ;

$$\mathcal{G}_{1(1,3)} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{pmatrix}; \quad \mathcal{G}_{1(2,3)} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 2 \end{pmatrix}; \quad \mathcal{G}_{1(3,3)} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Table 1 shows the type  $2^{\gamma}4^{\delta}$  of all these  $\mathcal{RM}_s(r, m)$  codes for  $m \leq 7$ .

The following proposition summarizes the parameters and properties of these  $\mathcal{RM}_s$  families of codes.

**Proposition 5 ([16],[17]).** *A quaternary linear Reed-Muller code  $\mathcal{RM}_s(r, m)$ , with  $m \geq 1$ ,  $0 \leq r \leq m$ , and  $0 \leq s \leq \lfloor \frac{m-1}{2} \rfloor$ , has the following parameters and properties:*

1. *the length is  $n = 2^{m-1}$ ;*
2. *the minimum Lee distance is  $d = 2^{m-r}$ ;*
3. *the number of codewords is  $2^k$ , where  $k = \sum_{i=0}^r \binom{m}{i}$ ;*
4. *the code  $\mathcal{RM}_s(r-1, m)$  is a subcode of  $\mathcal{RM}_s(r, m)$  for  $0 \leq r \leq m$ ;*
5. *the codes  $\mathcal{RM}_s(1, m)$  and  $\mathcal{RM}_s(m-2, m)$ , after the Gray map, are  $\mathbb{Z}_4$ -linear Hadamard and  $\mathbb{Z}_4$ -linear extended perfect codes, respectively;*
6. *the code  $\mathcal{RM}_s(r, m)$  is the dual code of  $\mathcal{RM}_s(m-1-r, m)$  for  $-1 \leq r \leq m$ .*

In the next two sections, for all these codes  $\mathcal{RM}_s(r, m)$  we will establish the dimension of the kernel, which will be denoted by  $k_{s(r,m)}$  instead of  $k_{\mathcal{RM}_s(r,m)}$ .

### 3 Kernel Dimensions for the $\mathcal{RM}_s$ Family with $s = 0$

In this section, we will compute the dimension of the kernel for the quaternary linear Reed-Muller codes in the  $\mathcal{RM}_s$  family with  $s = 0$ . As we have shown in Subsection 2.2, these codes can be constructed using the Plotkin construction.

Let  $\mathcal{C}$  be a quaternary linear code. The code  $2\mathcal{C}$  is obtained from  $\mathcal{C}$  by multiplying by two all codewords of  $\mathcal{C}$ . Note that if  $\mathcal{G}$  is a generator matrix of  $\mathcal{C}$ , then  $2\mathcal{G}$  is a generator matrix of  $2\mathcal{C}$ .

**Lemma 6.** *For all  $m \geq 1$  and  $r \in \{0, \dots, m-1\}$ ,  $2\mathcal{RM}_0(r+1, m) \subseteq \mathcal{RM}_0(r, m)$ .*

*Proof.* We proceed by induction. For  $m = 1$ , there are the zero, repetition and universe codes for  $r < 0$ ,  $r = 0$ , and  $r > 0$ , respectively. The lemma is true for these codes.

For  $m \geq 2$ , assume that the result is true. Let  $\mathcal{G}_{r-1,m-1}$ ,  $\mathcal{G}_{r,m-1}$  and  $\mathcal{G}_{r+1,m-1}$  be generator matrices of  $\mathcal{RM}_0(r-1, m-1)$ ,  $\mathcal{RM}_0(r, m-1)$  and  $\mathcal{RM}_0(r+1, m-1)$ , respectively. Using the Plotkin construction given by Definition 3 we obtain two new codes  $\mathcal{RM}_0(r, m)$  and  $\mathcal{RM}_0(r+1, m)$  with generator matrices

$$\mathcal{G}_{r,m} = \begin{pmatrix} \mathcal{G}_{r,m-1} & \mathcal{G}_{r,m-1} \\ 0 & \mathcal{G}_{r-1,m-1} \end{pmatrix}, \quad \mathcal{G}_{r+1,m} = \begin{pmatrix} \mathcal{G}_{r+1,m-1} & \mathcal{G}_{r+1,m-1} \\ 0 & \mathcal{G}_{r,m-1} \end{pmatrix},$$

respectively. Since  $2\mathcal{RM}_0(r + 1, m - 1) \subseteq \mathcal{RM}_0(r, m - 1)$ , the submatrix  $2(\mathcal{G}_{r+1, m-1} \ \mathcal{G}_{r+1, m-1})$  generates a code contained in the code generated by the submatrix  $(\mathcal{G}_{r, m-1} \ \mathcal{G}_{r, m-1})$ . The same argument can be used for the submatrices  $(0 \ \mathcal{G}_{r, m-1})$  and  $(0 \ \mathcal{G}_{r-1, m-1})$ . Thus the code  $2\mathcal{RM}_0(r + 1, m)$  generated by  $2\mathcal{G}_{r+1, m}$  is contained in the code  $\mathcal{RM}_0(r, m)$  generated by  $\mathcal{G}_{r, m}$ .  $\square$

Let  $W^{m-1}$  be the set of order four vectors  $\{w_1^{m-1}, \dots, w_m^{m-1}\}$  over  $\mathbb{Z}_4^{2^{m-1}}$  defined as follows:

$$\begin{aligned} w_1^{m-1} &= 1^{2^{m-1}} = \mathbf{1}, \\ w_i^{m-1} &= (0^{2^{m-i}} | 1^{2^{m-i}})^{2^{i-2}}, \text{ for } i \in \{2, \dots, m\}. \end{aligned}$$

Note that, for  $i \in \{2, \dots, m - 1\}$ , we have that  $(w_i^{m-2} | w_i^{m-2}) = w_{i+1}^{m-1}$ .

By Proposition 5, since the corresponding  $\mathbb{Z}_4$ -linear code of any  $\mathcal{RM}_s(1, m)$  is a Hadamard code,  $\mathbf{1} \in \mathcal{RM}_s(1, m)$  [13]. Moreover, for all  $r \geq 1$ , the vector  $\mathbf{1} \in \mathcal{RM}_s(r, m)$  by the inclusion property, and also belongs to the kernel of  $\mathcal{RM}_s(r, m)$  by Lemma 1.

**Lemma 7.** For all  $m \geq 2$  and  $r \in \{2, \dots, m\}$ ,  $W^{m-1}$  is a subset of  $\mathcal{RM}_0(r, m)$ .

*Proof.* It is clear that  $w_1^{m-1} = \mathbf{1} \in \mathcal{RM}_0(r, m)$  for  $r \geq 1$  and  $\mathbf{0} \in \mathcal{RM}_0(r, m)$  for  $r \geq 0$ . Hence, for  $m = 1$ ,  $W^0 = \{\mathbf{1}\}$  is a subset of  $\mathcal{RM}_0(1, 1)$ . For  $m \geq 2$ , the subsequent  $\mathcal{RM}_0(r, m)$  codes are obtained using the Plotkin construction given by Definition 3 as follows:

$$\mathcal{RM}_0(r, m) = \mathcal{PC}(\mathcal{RM}_0(r, m - 1), \mathcal{RM}_0(r - 1, m - 1)).$$

We proceed by induction on  $m$ . For  $m = 2$ , we have the set  $W^1 = \{w_1^1, w_2^1\}$  and the lemma is true, since  $\mathcal{RM}_0(r, 2)$  is the universe code for  $r \geq 2$ .

For  $m \geq 3$ , since  $\mathbf{0} \in \mathcal{RM}_0(r, m - 1)$  and  $\mathbf{1} \in \mathcal{RM}_0(r - 1, m - 1)$  for  $r \geq 2$ , then  $(\mathbf{0} | \mathbf{0} + \mathbf{1}) = w_2^{m-1} \in \mathcal{RM}_0(r, m)$  for  $r \geq 2$ . In general, if  $x \in \mathcal{RM}_0(r, m - 1)$ , then  $(x | x + \mathbf{0}) = (x | x) \in \mathcal{RM}_0(r, m)$ . Since  $w_i^{m-2} \in \mathcal{RM}_0(r, m - 1)$ , for  $r \geq 2$  and  $2 \leq i \leq m - 1$ , it is clear that  $(w_i^{m-2} | w_i^{m-2}) = w_{i+1}^{m-1} \in \mathcal{RM}_0(r, m)$ . Therefore,  $w_i^{m-1} \in \mathcal{RM}_0(r, m)$  for  $r \geq 2$  and  $1 \leq i \leq m$ .  $\square$

By Lemma 7, for all  $m \geq 2$  and  $r \in \{2, \dots, m\}$ , there is a generator matrix  $\mathcal{G}_{0(r, m)} = \begin{pmatrix} \mathcal{G}_\gamma \\ \mathcal{G}_\delta \end{pmatrix}$  of the code  $\mathcal{RM}_0(r, m)$  of type  $2^\gamma 4^\delta$ , such that  $W^{m-1}$  is a submatrix of  $\mathcal{G}_\delta$ , where  $\mathcal{G}_\gamma$  and  $\mathcal{G}_\delta$  are the  $\gamma$  and  $\delta$  generators of order two and four, respectively. In Proposition 10, for all  $m \geq 4$  and  $r \in \{2, \dots, m - 2\}$ , we will show that the kernel of  $\mathcal{RM}_0(r, m)$  is generated by the matrix

$$\begin{pmatrix} \mathcal{G}_\gamma \\ 2\mathcal{G}_\delta \\ W^{m-1} \end{pmatrix}. \tag{2}$$

**Lemma 8.** *Let  $\mathcal{C}_1$  and  $\mathcal{C}_2$  be two quaternary linear codes of length  $n$  with generator matrices  $\mathcal{G}_1$  and  $\mathcal{G}_2$ , respectively, such that  $\mathcal{C}_2 \subseteq \mathcal{C}_1$ . Let  $\mathcal{C} = \mathcal{PC}(\mathcal{C}_1, \mathcal{C}_2)$  of length  $2n$ . If  $x \in \mathcal{C}_1$  and  $y \in \mathcal{C}_2$ , then  $(x|x+y) \in \mathcal{K}(\mathcal{C})$  if and only if  $x \in \mathcal{K}(\mathcal{C}_1)$ ,  $2y * u \in \mathcal{C}_2$ , and  $2x * v \in \mathcal{C}_2$ , for all  $u \in \mathcal{G}_1$  and  $v \in \mathcal{G}_2$ .*

*Proof.* The codeword  $(x|x+y) \in \mathcal{K}(\mathcal{C})$  if and only if  $2(x|x+y) * (u|u) \in \mathcal{C}$  and  $2(x|x+y) * (0|v) \in \mathcal{C}$  for all  $u, v$  in  $\mathcal{G}_1, \mathcal{G}_2$ , respectively. That means  $(2x * u|2x * u + 2y * u) \in \mathcal{C}, \forall u \in \mathcal{G}_1$ , and  $(0|2x * v + 2y * v) \in \mathcal{C}, \forall v \in \mathcal{G}_2$ . That is,  $x \in \mathcal{K}(\mathcal{C}_1)$  and  $2y * u \in \mathcal{C}_2, \forall u \in \mathcal{G}_1$ , and  $2x * v + 2y * v \in \mathcal{C}_2, \forall v \in \mathcal{G}_2$ . Note that since  $\mathcal{C}_2 \subseteq \mathcal{C}_1$ , the condition  $2y * u \in \mathcal{C}_2 \forall u \in \mathcal{G}_1$  implies that  $2y * v \in \mathcal{C}_2 \forall v \in \mathcal{G}_2$ . Therefore,  $2x * v + 2y * v \in \mathcal{C}_2 \forall v \in \mathcal{G}_2$  is simplified to  $2x * v \in \mathcal{C}_2 \forall v \in \mathcal{G}_2$ .  $\square$

Note that if  $2y * u \in \mathcal{C}_2$  for all  $u \in \mathcal{G}_1$ , then  $y \in \mathcal{K}(\mathcal{C}_2)$ . Furthermore,  $x$  could not belong to  $\mathcal{C}_2$ , but if  $x \in \mathcal{C}_2$  then  $x \in \mathcal{K}(\mathcal{C}_2)$ .

**Proposition 9.** *For all  $m \geq 1$  and  $r \in \{0, 1, 2, m - 1, m\}$ , the corresponding  $\mathbb{Z}_4$ -linear code of  $\mathcal{RM}_0(r, m)$  is a binary linear code.*

*Proof.* For  $r = 0, r = m - 1$  and  $r = m$ , the corresponding  $\mathbb{Z}_4$ -linear codes of  $\mathcal{RM}_0(r, m)$  are the repetition, the even weight and the universe codes, respectively, which are binary linear codes. For  $r = 1$ , the corresponding  $\mathbb{Z}_4$ -linear code of  $\mathcal{RM}_0(r, m)$  is the binary linear Hadamard code [16],[17].

Finally, for  $r = 2$ , the  $\mathcal{RM}_0(2, m)$  code is constructed as  $\mathcal{PC}(\mathcal{RM}_0(2, m - 1), \mathcal{RM}_0(1, m - 1))$ . We proceed by induction on  $m$ . For  $m = 2$ , the code  $\mathcal{RM}_0(2, 2)$  is the universe code. For  $m \geq 3$ , we can assume that  $\phi(\mathcal{RM}_0(2, m - 1))$  and  $\phi(\mathcal{RM}_0(1, m - 1))$  are binary linear codes. The generator matrix of  $\mathcal{RM}_0(2, m)$  only have vectors of the form  $(x|x)$  and  $(0|y)$  for all  $x \in \mathcal{G}_{0(2, m-1)}$ ,  $y \in \mathcal{G}_{0(1, m-1)}$ . By Lemmas 6 and 8, since  $x \in \mathcal{K}(\mathcal{RM}_0(2, m - 1))$  and the only vector of order four in the generator matrix of  $\mathcal{RM}_0(1, m - 1)$  is  $\mathbf{1}$ , the vector  $(x|x) \in \mathcal{K}(\mathcal{RM}_0(2, m)), \forall x \in \mathcal{G}_{0(2, m-1)}$ . By Lemmas 6, 8 and the same argument, the vector  $(0|y) \in \mathcal{K}(\mathcal{RM}_0(2, m)), \forall y \in \mathcal{G}_{0(1, m-1)}$ . Thus all the vectors in the generator matrix of  $\mathcal{RM}_0(2, m)$  belong to the kernel of  $\mathcal{RM}_0(2, m)$ . Therefore, the corresponding  $\mathbb{Z}_4$ -linear code of  $\mathcal{RM}_0(2, m)$  is a binary linear code.  $\square$

Let  $A$  and  $B$  be two matrices. If  $B$  is a submatrix of  $A$ , then we will use  $A \setminus B$  to denote the matrix  $A$  without the rows of  $B$ . Recall that we will also use  $A$  and  $B$  to denote the set of row vectors of  $A$  and  $B$ , respectively. Moreover, if  $B \subseteq A$  then we will use  $A \setminus B$  to denote the set of row vectors in  $A$  which are not in  $B$ .

**Proposition 10.** *For all  $m \geq 4$  and  $r \in \{2, \dots, m - 2\}$ , the kernel of  $\mathcal{RM}_0(r, m)$  of type  $2^\gamma 4^\delta$  is the quaternary linear code generated by  $\begin{pmatrix} \mathcal{G}_\gamma \\ 2\mathcal{G}_\delta \\ W^{m-1} \end{pmatrix}$ , where  $\mathcal{G}_\gamma$  and  $\mathcal{G}_\delta$  are the  $\gamma$  and  $\delta$  generators of order two and four of  $\mathcal{RM}_0(r, m)$ , respectively.*

*Proof.* For  $m = 4$ , there is only the code  $\mathcal{RM}_0(2, 4)$ . By Proposition 9, the corresponding  $\mathbb{Z}_4$ -linear code of  $\mathcal{RM}_0(2, m)$  is a binary linear code. Since  $W^{m-1} \subseteq \mathcal{RM}_0(2, m)$  and  $\delta = m$ , the proposition is true for all the codes  $\mathcal{RM}_0(2, m)$ .

For  $m = 5$ , the code  $\mathcal{RM}_0(3, 5)$  is the first one under the Gray map, the binary code is nonlinear. In this case, we can compute its kernel and see that can be generated by  $\begin{pmatrix} \mathcal{G}_\gamma \\ 2\mathcal{G}_\delta \\ W^4 \end{pmatrix}$ .

By Lemma 7, there is a generator matrix for any  $\mathcal{RM}_0(r, m)$  with  $r \geq 2$ , which can be written as follows:

$$\begin{pmatrix} \mathcal{G}_\gamma \\ 2\mathcal{G}_\delta \\ \mathcal{G}_\delta \setminus W^{m-1} \\ W^{m-1} \end{pmatrix}.$$

For  $m > 5$ , assume that the lemma is true for  $m - 1$ . Let  $\mathcal{C}_1 = \mathcal{RM}_0(r, m - 1)$  and  $\mathcal{C}_2 = \mathcal{RM}_0(r - 1, m - 1)$  of types  $2^\gamma 4^\delta$  and  $2^{\gamma'} 4^{\delta'}$ , respectively. Let  $\mathcal{C} = \mathcal{RM}_0(r, m) = \mathcal{PC}(\mathcal{C}_1, \mathcal{C}_2)$  be the new code with a generator matrix of the form

$$\begin{pmatrix} \mathcal{G}_\gamma & \mathcal{G}_\gamma \\ 2\mathcal{G}_\delta & 2\mathcal{G}_\delta \\ \mathcal{G}_\delta \setminus W^{m-2} & \mathcal{G}_\delta \setminus W^{m-2} \\ W^{m-2} & W^{m-2} \\ 0 & \mathcal{G}_{\gamma'} \\ 0 & 2\mathcal{G}_{\delta'} \\ 0 & \mathcal{G}_{\delta'} \setminus W^{m-2} \\ 0 & W^{m-2} \end{pmatrix}$$

The vectors of order two are always in the kernel. By Lemma 8, the vectors that are not in the kernels of  $\mathcal{C}_1$  and  $\mathcal{C}_2$  can not be in the kernel of  $\mathcal{C}$ . This excludes the row vectors in  $(\mathcal{G}_\delta \setminus W^{m-2} \mid \mathcal{G}_\delta \setminus W^{m-2})$  and  $(0 \mid \mathcal{G}_{\delta'} \setminus W^{m-2})$ , and their linear combinations with the kernel of  $\mathcal{C}$ . Since  $W^{m-2} \subseteq \mathcal{K}(\mathcal{C}_1)$  and  $W^{m-2} \subseteq \mathcal{K}(\mathcal{C}_2)$ , the row vectors of the form  $(W^{m-2} \mid W^{m-2})$  are in  $\mathcal{K}(\mathcal{C})$ . For the vectors of the form  $(0 \mid W^{m-2})$ , we have two cases. By Lemmas 6 and 8,  $(\mathbf{0} \mid \mathbf{1}) \in \mathcal{K}(\mathcal{C})$ . On the other hand, the vectors  $w_i \in W^{m-2}$ ,  $2 \leq i \leq m - 1$ , have weight  $2^{m-3}$ . By Proposition 5,  $w_{min}(\mathcal{C}) = w_{min}(\mathcal{C}_2)$  and  $2w_{min}(\mathcal{C}) = w_{min}(\mathcal{C}_1)$ . For every vector  $y$  in the space generated by  $W^{m-2} \setminus \mathbf{1}$ , there is another vector of order four,  $u \in \mathcal{G}_1$ , such that the weight of  $2y * u$  is less than  $w_{min}(\mathcal{C})$ . Thus  $y \notin \mathcal{K}(\mathcal{C})$ . Finally,  $(W^{m-2} \mid W^{m-2}) \cup (\mathbf{0} \mid \mathbf{1}) = W^{m-1}$ .  $\square$

Note that the case  $r = 2$  is included in both Propositions 9 and 10. That is, the corresponding  $\mathbb{Z}_4$ -linear code of any  $\mathcal{RM}_0(2, m)$  is always a binary linear code and  $\mathcal{RM}_0(2, m)$  can be generated by a matrix of the form (2).

**Corollary 11.** *For all  $m \geq 1$  and  $0 \leq r \leq m$ , the dimension of the kernel of  $\mathcal{RM}_0(r, m)$  of type  $2^\gamma 4^\delta$  is*

$$k_{0(r,m)} = \begin{cases} \gamma + 2\delta & \text{if } r \in \{0, 1, m - 1, m\} \\ \gamma + \delta + m & \text{if } r \in \{2, \dots, m - 2\}. \end{cases}$$

*Proof.* Straightforward from Propositions 9 and 10. □

Note that, the kernel, as a binary linear code, is generated by

$$\begin{pmatrix} \phi(\mathcal{G}_\gamma) \\ \phi(2\mathcal{G}_\delta) \\ \phi(W^{m-1}) \end{pmatrix},$$

where all these vectors are linear independent over  $\mathbb{Z}_2^m$ .

## 4 Kernel Dimensions for the $\mathcal{RM}_s$ Families

In this section, the general case when  $s > 0$  is studied. We will establish the dimension of the kernel for any quaternary linear Reed-Muller code in these  $\mathcal{RM}_s$  families. This invariant, the kernel dimension, will help us to the classification of these codes.

As we have shown in Subsection 2.2, these quaternary linear Reed-Muller codes  $\mathcal{RM}_s(r, m)$  can be obtained using the Plotkin construction, except when  $m$  is odd and  $s = \frac{m-1}{2}$ . In this case, they are obtained using the BQ-Plotkin construction. Note that some of these codes could be constructed using any of these two constructions.

**Theorem 12.** *For all  $m \geq 1$ ,  $0 \leq r \leq m$ , and  $0 \leq s \leq \lfloor \frac{m-1}{2} \rfloor$ , the dimension of the kernel of  $\mathcal{RM}_s(r, m)$  of type  $2^\gamma 4^\delta$  is*

1.  $k_{s(0,m)} = 1$ ,  $k_{s(m-1,m)} = 2^m - 1$ ,  $k_{s(m,m)} = 2^m$ .
2. If  $s = 0$ ,

$$k_{0(r,m)} = \begin{cases} \gamma + 2\delta & \text{if } r = 1 \\ \gamma + \delta + m & \text{if } r \in \{2, \dots, m-2\}. \end{cases}$$

3. If  $s = 1$ ,  $k_{1(r,m)} = \gamma + \delta + 2$  for all  $r \in \{1, 2, \dots, m-2\}$ .
4. If  $s \geq 2$ ,  $k_{s(r,m)} = \gamma + \delta + 1$  for all  $r \in \{1, 2, \dots, m-2\}$ , except the case  $k_{2(2,5)} = \gamma + \delta + 2 = 11$ .

*Proof.* It is straightforward to see that  $k_{s(0,m)} = 1$ ,  $k_{s(m-1,m)} = 2^m - 1$ , and  $k_{s(m,m)} = 2^m$ , because  $\phi(\mathcal{RM}_s(0, m))$ ,  $\phi(\mathcal{RM}_s(m-1, m))$ , and  $\phi(\mathcal{RM}_s(m, m))$  are the repetition, the even weight, and the universe codes, respectively.

The case  $s = 0$  is proved in Corollary 11. The cases  $s = 1$  and  $s \geq 2$  can be proved using similar arguments to that for the case  $s = 0$ . When  $s = 1$  there are only two generators of order four in the kernel of  $\mathcal{RM}_1(r, m)$ ,  $w_1^{m-1} = \mathbf{1}$  and  $w_2^{m-1}$ . When  $s \geq 2$  there is only one generator of order four  $w_1^{m-1} = \mathbf{1}$  in the kernel of  $\mathcal{RM}_s(r, m)$ , except for the code  $\mathcal{RM}_2(2, 5)$ . Since  $\phi(\mathcal{RM}_2(2, 5))$  and  $\phi(\mathcal{RM}_2(1, 5))$  are equivalent  $k_{2(2,5)} = 11 = \gamma + \delta + 2$ . □

Note that Theorem 12 includes the previous results about the kernel dimension for  $\mathbb{Z}_4$ -linear Hadamard and  $\mathbb{Z}_4$ -linear extended 1-perfect codes [6],[13],[15] or (1). Table 1 shows the type  $2^\gamma 4^\delta$  and the dimension of the kernel of all these  $\mathcal{RM}_s(r, m)$  codes for  $m \leq 7$ .

**Table 1.** Type  $2^{\gamma}4^{\delta}$  and kernel dimension  $k_{s(r,m)}$  for all  $\mathcal{RM}_s(r,m)$  codes with  $m \leq 7$ , showing them in the form  $(\gamma, \delta) k_{s(r,m)}$

$m \backslash s \backslash r$	0	1	2	3	4	5	6	7	
1	0	(1,0) 1	(0,1) 2						
2	0	(1,0) 1	(1,1) 3	(0,2) 4					
3	0	(1,0) 1	(2,1) 4	(1,3) 7	(0,4) 8				
	1	(1,0) 1	(0,2) 4	(1,3) 7	(0,4) 8				
4	0	(1,0) 1	(3,1) 5	(3,4) 11	(1,7) 15	(0,8) 16			
	1	(1,0) 1	(1,2) 5	(1,5) 8	(1,7) 15	(0,8) 16			
5	0	(1,0) 1	(4,1) 6	(6,5) 16	(4,11) 20	(1,15) 31	(0,16) 32		
	1	(1,0) 1	(2,2) 6	(2,7) 11	(2,12) 16	(1,15) 31	(0,16) 32		
	2	(1,0) 1	(0,3) 4	(2,7) 11	(0,13) 14	(1,15) 31	(0,16) 32		
6	0	(1,0) 1	(5,1) 7	(10,6) 22	(10,16) 32	(5,26) 37	(1,31) 63	(0,32) 64	
	1	(1,0) 1	(3,2) 7	(4,9) 15	(4,19) 25	(3,27) 32	(1,31) 63	(0,32) 64	
	2	(1,0) 1	(1,3) 5	(2,10) 13	(2,20) 23	(1,28) 30	(1,31) 63	(0,32) 64	
7	0	(1,0) 1	(6,1) 8	(15,7) 29	(20,22) 49	(15,42) 64	(6,57) 70	(1,63) 127	(0,64) 128
	1	(1,0) 1	(4,2) 8	(7,11) 20	(8,28) 38	(7,46) 55	(4,58) 64	(1,63) 127	(0,64) 128
	2	(1,0) 1	(2,3) 6	(3,13) 17	(4,30) 35	(3,48) 52	(2,59) 62	(1,63) 127	(0,64) 128
	3	(1,0) 1	(0,4) 5	(3,13) 17	(0,32) 33	(3,48) 52	(0,60) 61	(1,63) 127	(0,64) 128

The next theorem proves that there are at least  $\lfloor \frac{m-1}{2} \rfloor$  nonequivalent binary codes with the same parameters as the code  $RM(r,m)$ .

**Lemma 13.** *Given two codes  $\mathcal{RM}_s(r,m)$  and  $\mathcal{RM}_{s'}(r,m)$  of type  $2^{\gamma}4^{\delta}$  and  $2^{\gamma'}4^{\delta'}$  respectively, such that  $s < s'$ , we have that  $\gamma + \delta > \gamma' + \delta'$ , except one case: if  $m$  is odd,  $r$  even,  $s = \frac{m-3}{2}$ , and  $s' = \frac{m-1}{2}$ , then  $\gamma + \delta = \gamma' + \delta'$ .*

*Proof.* When  $\mathcal{RM}_s(r,m)$  and  $\mathcal{RM}_{s'}(r,m)$  are obtained using the Plotkin construction, it is easy to see that  $\gamma + \delta > \gamma' + \delta'$ .

If  $m$  is odd and  $s' = \frac{m-1}{2}$ , the code  $\mathcal{RM}_{s'}(r,m)$  is obtained using the BQ-Plotkin construction. Without loss of generality, we can assume that  $s = s' - 1$  and  $\mathcal{RM}_s(r,m)$  is obtained using the Plotkin construction. Using the recursive definition of  $\gamma = \gamma_{s(r,m)}$ ,  $\delta = \delta_{s(r,m)}$ ,  $\gamma' = \gamma_{s'(r,m)}$ , and  $\delta = \delta_{s'(r,m)}$ , we have that  $\gamma_{s(r,m)} + \delta_{s(r,m)} - \gamma_{s'(r,m)} - \delta_{s'(r,m)} = \gamma_{s'-1(r-1,m-2)}$ . Hence, if  $\gamma_{s'-1(r-1,m-2)} \neq 0$  then  $\gamma_{s(r,m)} + \delta_{s(r,m)} > \gamma_{s'(r,m)} + \delta_{s'(r,m)}$ , and if  $\gamma_{s'-1(r-1,m-2)} = 0$  then  $\gamma_{s(r,m)} + \delta_{s(r,m)} = \gamma_{s'(r,m)} + \delta_{s'(r,m)}$ , which is when  $r$  is even.  $\square$

**Theorem 14.** *For all  $m \geq 4$  and  $r = 1$ , there are at least  $\lfloor \frac{m-1}{2} \rfloor$  nonequivalent binary codes with the same parameters as the code  $RM(1,m)$ .*

*For all  $m \geq 4$  and  $2 \leq r \leq m - 2$ , there are at least  $\lfloor \frac{m+1}{2} \rfloor$  nonequivalent binary codes with the same parameters as the code  $RM(r,m)$ , except when  $m$  is odd, and  $r$  is even. In this case, there are at least  $\frac{m-1}{2}$  nonequivalent binary codes with the same parameters as the code  $RM(r,m)$ .*

*Proof.* For  $r = 1$ , the result was proved in [13]. For  $2 \leq r \leq m - 2$ , the proof is consequence of Theorem 12 and Lemma 13.  $\square$

## 5 Conclusions

In a recent paper [17], new families of quaternary linear codes, the  $\mathcal{RM}_s(r, m)$  codes, are constructed in such a way that, after the Gray map, the  $\mathbb{Z}_4$ -linear codes fulfill the same properties and fundamental characteristics as the binary linear Reed-Muller codes. In this paper, a structural invariant for binary codes, the kernel dimension, is used to classify these new families of codes. Using a recursive construction, we give the generator matrices of the kernel and compute the exact values of the kernel dimension for all the feasible values of  $s$ ,  $r$  and  $m$ . This invariant allows us to classify all the codes except when  $m$  is odd,  $m \geq 5$ , and  $r$  is even. In a further research we will also compute the rank, another structural invariant for binary codes, and give a complete classification of these families of codes.

## References

1. Bauer, H., Ganter, B., Hergert, F.: Algebraic techniques for nonlinear codes. *Combinatorica* 3, 21–33 (1983)
2. Borges, J., Fernández, C., Phelps, K.T.: Quaternary Reed-Muller codes. *IEEE Trans. Inform. Theory* 51(7), 2686–2691 (2005)
3. Borges, J., Fernández-Córdoba, C., Phelps, K.T.: ZRM codes. *IEEE Trans. Inform. Theory* 54(1), 380–386 (2008)
4. Pernas, J., Pujol, J., Villanueva, M.: Codes over  $\mathbb{Z}_4$ . A Magma package. Universitat Autònoma de Barcelona (2008), <http://www.ccg.uab.cat>
5. Borges, J., Phelps, K.T., Rifà, J., Zinoviev, V.A.: On  $\mathbb{Z}_4$ -linear Preparata-like and Kerdock-like codes. *IEEE Trans. Inform. Theory* 49(11), 2834–2843 (2003)
6. Borges, J., Phelps, K.T., Rifà, J.: The rank and kernel of extended 1-perfect  $\mathbb{Z}_4$ -linear and additive non- $\mathbb{Z}_4$ -linear codes. *IEEE Trans. Inform. Theory* 49(8), 2028–2034 (2003)
7. Cannon, J.J., Bosma, W. (eds.): *Handbook of Magma Functions*, Edition 2.13, 4350 p. (2006)
8. Fernández-Córdoba, C., Pujol, J., Villanueva, M.: On rank and kernel of  $\mathbb{Z}_4$ -linear codes. In: Barbero, A. (ed.) *ICMCTA 2008*. LNCS, vol. 5228, pp. 46–55. Springer, Heidelberg (2008)
9. Fernández-Córdoba, C., Pujol, J., Villanueva, M.:  $\mathbb{Z}_2\mathbb{Z}_4$ -linear codes: rank and kernel. *Discrete Applied Mathematics* (submitted, 2008) (arXiv:0807.4247)
10. Hammons, A.R., Kumar, P.V., Calderbank, A.R., Sloane, N.J.A., Solé, P.: The  $\mathbb{Z}_4$ -linearity of Kerdock, Preparata, Goethals and related codes. *IEEE Trans. Inform. Theory* 40, 301–319 (1994)
11. Hou, X.-D., Lahtonen, J.T., Koponen, S.: The Reed-Muller code  $R(r, m)$  is not  $\mathbb{Z}_4$ -linear for  $3 \leq r \leq m - 2$ . *IEEE Trans. Inform. Theory* 44, 798–799 (1998)
12. Huffman, W.C., Pless, V.: *Fundamentals of Error-Correcting Codes*. Cambridge University Press, Cambridge (2003)

13. Krotov, D.S.:  $\mathbb{Z}_4$ -linear Hadamard and extended perfect codes. In: International Workshop on Coding and Cryptography, Paris, France, January 8-12, pp. 329-334 (2001)
14. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes. North-Holland Publishing Company, Amsterdam (1977)
15. Phelps, K.T., Rifà, J., Villanueva, M.: On the additive ( $\mathbb{Z}_4$ -linear and non- $\mathbb{Z}_4$ -linear) Hadamard codes: rank and kernel. IEEE Trans. Inform. Theory 52(1), 316-319 (2006)
16. Pujol, J., Rifà, J., Solov'eva, F.I.: Quaternary plotkin constructions and quaternary reed-muller codes. In: Boztaş, S., Lu, H.-F.(F.) (eds.) AAEECC 2007. LNCS, vol. 4851, pp. 148-157. Springer, Heidelberg (2007)
17. Pujol, J., Rifà, J., Solov'eva, F.I.: Construction of  $\mathbb{Z}_4$ -linear Reed-Muller codes. IEEE Trans. Inform. Theory (to appear, 2008)
18. Solov'eva, F.I.: On  $\mathbb{Z}_4$ -linear codes with parameters of Reed-Muller codes. Problems of Inform. Trans. 43(1), 26-32 (2007)
19. Wan, Z.-X.: Quaternary codes. World Scientific Publishing Co. Pte. Ltd., Singapore (1997)

# Coding-Based Oblivious Transfer

Kazukuni Kobara<sup>1</sup>, Kirill Morozov<sup>1</sup>, and Raphael Overbeck<sup>2,\*</sup>

<sup>1</sup> RCIS, AIST

Akihabara Daibiru, room 1102

1-18-13 Sotokanda, Chiyoda-ku

Tokyo 101-0021 Japan

{k-kobara,kirill.morozov}@aist.go.jp

<sup>2</sup> EPFL - I&C - ISC - LASEC

Station 14 - Building INF

CH-1015 Lausanne

Switzerland

overbeck@cdc.informatik.tu-darmstadt.de

**Abstract.** We present protocols for two flavors of oblivious transfer (OT): the Rabin and 1-out-of-2 OT based on the assumptions related to security of the McEliece cryptosystem and two zero-knowledge identification (ZKID) schemes, Stern's from Crypto '93 and Shamir's from Crypto '89, which are based on syndrome decoding and permuted kernels, respectively. This is a step towards diversifying computational assumptions on which OT – cryptographic primitive of central importance – can be based.

As a by-product, we expose new interesting applications for both ZKID schemes: Stern's can be used for proving correctness of McEliece encryption, while Shamir's – for proving that some matrix represents a permuted subcode of a given code.

Unfortunately, it turned out to be difficult to reduce the sender's security of both schemes to a hard problem, although the intuition suggests a successful attack may allow to solve some long-standing problems in coding theory.

**Keywords:** Oblivious transfer, coding-based cryptography, McEliece cryptosystem, permuted kernel problem.

## 1 Introduction

Oblivious transfer (OT) [10, 23, 28] is an important cryptographic primitive which implies secure two-party computation [12, 16]. OT guarantees a transmission from a sender to a receiver with partial erasure of the input, which can happen in two manners. When the whole input is erased with some fixed probability (independently of the player's control), we have an analog of the erasure channel, or Rabin OT. When the sender has two inputs and one of them is received (while the other is erased) according to the receiver's choice (and the

---

\* Part of this work was done at the Cryptography and Computer Algebra Group, TU-Darmstadt, Germany. Part of this work was funded by the DFG.

sender does not learn this choice), we have 1-out-of-2 OT. In fact, these two flavors of OT were shown to be equivalent [7].

A number of complexity assumptions were used to construct OT: generic, e.g., enhanced trapdoor permutations [10, 11, 14], and specific, e.g., factoring [23], Diffie-Hellman [1, 3, 20], N'th or Quadratic Residuosity and Extended Riemann Hypothesis [15].

**Our contribution.** We present two coding-based computationally secure constructions. The Rabin OT protocol is based on the McEliece encryption [19] where a public-key is constructed from the permuted concatenation of the standard McEliece public key and a random matrix. The receiver will construct this public key and prove its correctness using the Shamir's zero-knowledge identification (ZKID) scheme [25] based on permuted kernel problem (PKP). The sender will prove that the input is encrypted using the error vector of the appropriate weight using the Stern's ZKID scheme [26] based on general syndrome decoding.

We note that these new applications of the two ZKID schemes can be of independent interest in coding-based protocols. For instance, combining McEliece encryption with Stern's ZKID yields the verifiable McEliece encryption (for verifiable encryption and its applications see, e.g., [4]).

Unfortunately, in the above protocol, even the honest-but-curious receiver can reduce the probability of erasure. We show that this can be fixed by applying the reduction [7]. In fact, we show that this reduction can be used for such a weaker version of Rabin OT. However, the whole construction becomes involved and we end up implementing 1-out-of-2 OT on the way. Hence, we present a generalization of the above protocol which implements 1-out-of-2 OT using the presented techniques.

The security of both protocols is based on the assumption related to security of the McEliece PKC – indistinguishability of permuted generating matrix of a Goppa code from random and bounded distance decoding – and, in addition, the assumptions underlying the used ZKID schemes. We also employ commitment schemes.

Both constructions share the same shortcoming: it turned out to be difficult to reduce the sender's security to a hard decoding problem. Shortly speaking, the intuition suggests that a successful attack would require either efficient list decoding algorithm for Goppa codes, or extending those codes with random columns while still retaining a good error correcting capability.

**Related Work.** The work [9] presents 1-out-of-2 Bit OT protocol based exclusively on the McEliece PKC related assumptions. Its efficiency is comparable to our 1-out-of-2 String OT protocol, however it provides a stronger security guarantee for the receiver: it is unconditional as compared to computational in our case.

**Organization.** In Section 2, we briefly introduce our security definitions, assumptions, and the main ingredients for our constructions. The reduction from

a weak version of Rabin OT to the original Rabin OT is presented in Section 3. Section 4 introduces our Rabin OT construction, while our 1-out-of-2 OT protocol is sketched in Section 5.

## 2 Preliminaries

In our definitions, the players are bounded to run in probabilistic polynomial time in a security parameter  $n$ .

For vectors, summation is component-wise in the corresponding field, unless stated otherwise. Computational indistinguishability is denoted by “ $\stackrel{c}{\equiv}$ ”.

### 2.1 Security Definitions

Informally, Rabin (string) oblivious transfer is the trusted erasure channel from the sender  $\text{Sen}$  to the receiver  $\text{Rec}$  with fixed erasure probability  $Q^H = 1 - \mathcal{P}$  and a bit-vector  $\mathbf{b} \in \mathbb{F}_2^k$  as input. The malicious sender  $\widetilde{\text{Sen}}$  has no knowledge on the output, while the malicious receiver  $\widetilde{\text{Rec}}$  cannot learn the erased input.

We denote by a *View* of the player all the messages that he sent and received during the protocol as well as his local randomness. Let the binary random variable  $E$  (which indicates the fact of erasure and whose outcome is available to  $\text{Rec}$ ) is equal to 0 with probability  $\mathcal{P}$ . For the sake of simplicity, in the expressions for views, we omit most of the variables which are the same on both sides of equality.

**Definition 2.1.** *A two-party protocol is said to securely implement Rabin OT, if  $\text{Sen}$  gets as input a  $k$ -bit vector  $\mathbf{m}$  and the following conditions are satisfied:*

- *Completeness:* When  $\text{Sen}$  and  $\text{Rec}$  follow the protocol, if  $E = 0$ , then  $\text{Rec}$  outputs  $\mathbf{m}$ , otherwise he outputs “erasure”.
- *Sender’s security:*  $\forall \mathbf{m}' \neq \mathbf{m}, \mathbf{m}' \in \mathbb{F}_2^k$  :  
 $\text{View}_{\widetilde{\text{Rec}}}(\mathbf{m}|E = 1) \stackrel{c}{\equiv} \text{View}_{\widetilde{\text{Rec}}}(\mathbf{m}'|E = 1)$ .
- *Receiver’s security:*  $\text{View}_{\widetilde{\text{Sen}}}(E = 0) \stackrel{c}{\equiv} \text{View}_{\widetilde{\text{Sen}}}(E = 1)$ .

The definition of  $\gamma$ -gap Rabin OT is analogous to the above, but  $\widetilde{\text{Rec}}$  can decrease the erasure probability from his point of view by  $\gamma$ . This probability is denoted by  $Q = 1 - \mathcal{P} - \gamma$ . Let the binary random variable  $\widetilde{E}$  which indicates the fact of erasure ( $\widetilde{E} = 1$ ) or not for  $\widetilde{\text{Rec}}$  be equal to 1 with probability  $Q$ .

**Definition 2.2.** *A two-party protocol is said to securely implement  $\gamma$ -gap Rabin OT, if Definition 2.1 holds except that the sender’s security condition is replaced with:*

$$\forall \mathbf{m}' \neq \mathbf{m}, \mathbf{m}' \in \mathbb{F}_2^k : \text{View}_{\widetilde{\text{Rec}}}(\mathbf{m}|\widetilde{E} = 1) \stackrel{c}{\equiv} \text{View}_{\widetilde{\text{Rec}}}(\mathbf{m}'|\widetilde{E} = 1).$$

In the other flavor of oblivious transfer, 1-out-of-2 String OT,  $\text{Sen}$  inputs two  $a$ -bit vectors  $\mathbf{b}_0, \mathbf{b}_1$ .  $\text{Rec}$  obtains one of them according to his choice  $c \in \{0, 1\}$ .  $\widetilde{\text{Sen}}$  is unable to learn  $c$ , while  $\widetilde{\text{Rec}}$  remains ignorant about  $\mathbf{b}_{1-c}$ .

## 2.2 Assumptions

The security of all schemes we present in this paper is based on the assumption, that the following problems are hard in the average case (i.e. all but a negligible fraction of random instances of these problems are infeasible to solve):

**Definition 2.3.** *In the following let all matrices and vectors be over  $\mathbb{F}_q$ .*

- (i) *Given a  $k \times n$  matrix, decide if its row-space is within a Goppa code or was generated at random. (Goppa-code-distinguishing Problem)*
- (ii) *Given a (random)  $[n, k]$  code generated by the matrix  $\mathbf{G}^{\text{pub}}$ , a word  $\mathbf{c}$  and an integer  $w$ , find  $\mathbf{e}$  of Hamming weight at most  $w$  such that  $\mathbf{c} = \mathbf{m}\mathbf{G}^{\text{pub}} + \mathbf{e}$  for some  $\mathbf{m}$ . (General Syndrome Decoding)*
- (iii) *Given a (random)  $[n, k, d]$  code generated by the matrix  $\mathbf{G}^{\text{pub}}$ , find a code-word of weight  $\leq w$  in that code (Finding low weight words).*
- (iv) *Given a random  $[n, k]$  code and a random permuted subcode of dimension  $l < k$ , find the permutation. (Permuted Kernel Problem)*

Only Problems (ii) – (iv) are known to be  $\mathcal{NP}$ -hard in the general case [25, 26]. The coding theoretic problems (ii) and (iii) seem to be the hardest, if  $w$  is close to the Gilbert-Varshamov (GV) bound (see, e.g. [18, Ch. 17, Thm. 30]).

## 2.3 Tools

We shortly recall the main ingredients of our scheme: the McEliece PKC, the zero-knowledge identification protocols (ZKID) by Stern and Shamir connected to coding theory, and Crépeau’s protocol for 1-out-of-2 OT based on Rabin OT.

**McEliece’s public key encryption scheme** [19] works as follows: Upon input of the system parameters  $m, t$ , the key generation algorithm outputs the secret key consisting of three matrices:  $(\mathbf{S}, \mathbf{G}, \mathbf{P})$ , where  $\mathbf{G} \in \mathbb{F}_2^{k \times n}$  is a canonical generator matrix of an  $[n, k \geq n - mt, 2t + 1]$  binary irreducible Goppa code,  $\mathbf{S} \in \mathbb{F}_2^{k \times k}$  is non-singular and  $\mathbf{P} \in \mathbb{F}_2^{n \times n}$  is a permutation matrix. The corresponding public key is  $(\mathbf{G}^{\text{pub}} = \mathbf{S}\mathbf{G}\mathbf{P}, t)$ . To encrypt a message  $\mathbf{m} \in \mathbb{F}_2^k$  the sender chooses a random binary vector  $\mathbf{e}$  of length  $n$  and Hamming weight  $t$  and computes the ciphertext  $\mathbf{c} = \mathbf{m}\mathbf{G}^{\text{pub}} + \mathbf{e}$ . The secret key holder now can recover  $\mathbf{m}$  from  $\mathbf{c}$  using his secret key.

For properly chosen parameters, the McEliece PKC is secure [5] and there exist conversions to obtain CCA2 security [17]. For such variants, or if only random strings are encrypted,  $\mathbf{G}^{\text{pub}}$  can be chosen to be *systematic* (i.e. with the  $k$ -dimensional identity matrix  $\text{Id}_k$  in the first  $k$  columns), as we will do in the following. This reduces space needed to store  $\mathbf{G}^{\text{pub}}$ .

The size of the ciphertexts can be reduced to  $n - k$  if the message is represented by  $\mathbf{e}$ . This is known as the Niederreiter PKC, compare [24]. In the latter case (e.g. if a hash of  $\mathbf{e}$  serves as a random seed or key for a symmetric encryption scheme)

it is sufficient to send the syndrome  $\mathbf{e}(\mathbb{G}^{\text{pub}})^\perp$  as ciphertext, where  $(\mathbb{G}^{\text{pub}})^\perp$  refers to the systematic check matrix of  $\mathbb{G}^{\text{pub}}$ .

**Stern's ZKID** [26] has a check matrix  $\mathbf{H} \in \mathbb{F}_q^{n \times (n-k)}$  and an integer  $w$  as system parameters. A user's identity  $\mathbf{s}$  is computed from the user's secret, a vector  $\mathbf{e} \in \mathbb{F}_q^n$  of Hamming weight  $w$ :  $\mathbf{s} = \mathbf{e}\mathbf{H}$ . By Stern's 3-round zero-knowledge protocol, the secret key holder can prove his knowledge of  $\mathbf{e}$  using two blending factors: a permutation and a random vector. However, a dishonest prover not knowing  $\mathbf{e}$  can cheat the verifier in the protocol with probability  $2/3$ . Thus, the protocol has to be run several times to detect cheating provers. Computing  $\mathbf{e}$  from  $\mathbf{s}$  is solving Problem (ii) from Definition 2.3. The communication cost is about  $n(1 + \log_2(n)) \log_2(q)$  plus three times the size of the employed commitments (e.g. a hash function).

**Shamir's Permuted Kernel ZKID** [25] works quite similarly, i.e. it has a check matrix  $\mathbf{H} \in \mathbb{F}_q^{n \times (n-k)}$  and an integer  $l$  as system parameters. (Shamir proposed to use  $l = 1$  and  $q$  to be a large prime. However, taking  $q$  small and  $l < (n - k)$  works as well [27].) The user's identity  $\mathbf{K} \in \mathbb{F}_q^{l \times n}$  is computed from the user's secret, a permutation  $\Pi$  as follows:  $\mathbf{K}$  is taken at random from the right kernel of  $\Pi\mathbf{H}$ . In the following we can view  $\mathbf{K}$  as an  $n$ -vector over  $\mathbb{F}_{q^l}$ . By Shamir's 5-round zero-knowledge protocol, the secret key holder can prove his knowledge of  $\Pi$  using two blending factors: a permutation and a random  $n$ -vector over  $\mathbb{F}_{q^l}$ . However, a dishonest prover not knowing  $\Pi$  can cheat the verifier in the protocol with probability  $(q^l + 1)/(2 \cdot q^l)$ . Thus, the protocol has to be repeated several times to detect cheating provers. Computing  $\Pi$  from  $\mathbf{K}$  is solving Problem (iv) from Definition 2.3. The communication cost is about  $n(l + \log_2(n)) \log_2(q)$  plus two times the size of the commitments. See [22] for the practical security analysis.

**Crépeau's protocol** [7] allows us to build a 1-out-of-2 OT from a Rabin OT and a hash function  $h$ : In a first stage,  $N$  random messages  $r_i$  are sent to the receiver by a Rabin OT with erasure (receiving) probability  $\mathcal{Q}$  ( $\mathcal{P}$ ). Now,  $K$  is chosen such that  $K < \mathcal{P}N = (1 - \mathcal{Q})N < 2K < N$ , i.e. the receiver obtains at least  $K$  and at most  $2K - 1$  of the random messages  $r_i$ . Then, the receiver sends two disjoint sets  $\mathcal{I}, \mathcal{J} \subseteq \{1, \dots, N\}$  of  $K$  indices to the sender, such that one of the sets contains only indices of not erased messages  $r_i$ . For the 1-out-of-2 OT, the messages  $\mathbf{m}_0, \mathbf{m}_1$  are encrypted as  $\mathbf{c}_0 = \mathbf{m}_0 + h((r_i)_{i \in \mathcal{I}})$  and  $\mathbf{c}_1 = \mathbf{m}_1 + h((r_j)_{j \in \mathcal{J}})$ . Since the receiver knows either the set  $(r_i)_{i \in \mathcal{I}}$  or  $(r_j)_{j \in \mathcal{J}}$ , he obtains exactly one of the messages from  $\mathbf{c}_0$  and  $\mathbf{c}_1$ . Crépeau's protocol fails with some probability which is negligible in  $N$  and can easily be computed.

**Commitment scheme.** This protocol allows a committer to transmit an evidence (called *commitment*) of a message to the verifier with possibility to reveal the message later. The committer cannot learn the message before revealing,

while the verifier cannot change his mind by opening a different message. In this work, we use a commitment functionality abstracting from its implementation. For details on commitment schemes, see [8] and the references therein.

### 3 Reducing the Gap in Rabin OT

We show that the Crépeau's protocol can be used to reduce the Gap Rabin OT to the 1-out-of-2 OT and thus to the original Rabin OT.

**Theorem 3.1.**  *$\gamma$ -Gap Rabin OT with  $\gamma = Q^H - Q^A$  is equivalent to Rabin OT, if for some  $K' > 0$ :  $K' < 1 - Q^H \leq 1 - Q^A < 2K' < 1$ .*

*Proof (Sketch).* Consider the Crépeau's protocol as described in the previous section and take  $K = K'N$ . It is easy to check that this protocol works, i.e., the sender is very likely to find enough received messages for one set, while the other set is very likely to contain at least one erasure.

### 4 Rabin Oblivious Transfer

Our scheme implementing Rabin OT with erasure probability  $1 - \mathcal{P}$  consists of two phases: initialization and transmission. The first one is used for key generation, where a Goppa code is concatenated with a random code and used as substitute for the secret code in the McEliece PKC, see Algorithm 4.1. To ensure correct generation of the public key, we use a trusted third party (TTP) in Algorithm 4.1, which can be omitted as we show in Section 4.2. In the

---

#### Algorithm 4.1. Key generation

---

**Input:** Security parameters  $m, t, t', l \in \mathbb{N}$ .

**Receiver:** Set  $n = 2^m, k = 2^m - mt$ . Generate a McEliece PKC key pair with security parameters  $m, t$ . Let  $(S, G, P)$  be the secret key with public key  $(G^{\text{pub}} = SGP, t)$ . Send  $G^{\text{pub}}$  to the TTP.

**TTP:** Generate a random matrix  $G' \in \mathbb{F}_2^{k \times l}$  and a  $(n+l) \times (n+l)$  random permutation matrix  $P'$ . Publish the systematic matrix  $O^{\text{pub}}$  generating the same  $[n+l, k]$  code as  $[G^{\text{pub}} | G'] P'$ .

---

transmission phase, see Algorithm 4.2, en- and decryption work like in the McEliece PKC. The difference lies in the modified public key, which ensures, that the receiver cannot decrypt all valid ciphertexts. The time complexity for Algorithm 4.2 is  $\mathcal{O}(n \cdot k + n \cdot m \cdot t^2)$  operations [6]. The size of the ciphertexts is  $n + l$ , but as mentioned in Section 2.3, this can be reduced to  $n + l - k$  by encoding the message into the error vector  $\mathbf{e}$ .

Note that if  $O^{\text{pub}}$  is re-used in the different instances of Algorithm 4.2, a security problem might arise when composing such instances, see discussion in Appendix A.2. For the sake of simplicity of our proofs, we henceforth assume that  $O^{\text{pub}}$  is generated each time anew.

**Algorithm 4.2.** Transmission

**Input:** The security parameters  $m, l, t'$  and a  $k$ -bit message  $\mathbf{m}$ .

**Encryption:**

Obtain the receiver's public key  $\mathbf{O}^{\text{pub}}$ .

Generate a random vector  $\mathbf{e}$  of weight  $t'$  and length  $2^m + l$ .

Compute the ciphertext  $\mathbf{c} = \mathbf{m}\mathbf{O}^{\text{pub}} + \mathbf{e}$ .

Send  $\mathbf{c}$  to the receiver.

Following Stern's protocol with system parameters  $\mathbf{O}^{\text{pub}}$  and  $t'$ , send a zero knowledge proof of knowledge **Proof** for the public key  $\mathbf{c}$  and secret key  $\mathbf{e}$  to the receiver.

**Decryption:**

Verify **Proof**.

Set  $(\mathbf{c}_1, \mathbf{c}_2) = \mathbf{c}(\mathbf{P}')^{-1}$ , where  $\mathbf{c}_1$  is an  $n$ -bit vector.

Try to apply the error correction algorithm for  $\mathbf{G}$  to  $\mathbf{c}_1\mathbf{P}^{-1}$  in order to obtain  $\mathbf{m}$ .

**if** (previous step fails) **or**  $(\mathbf{m}\mathbf{O}^{\text{pub}} + \mathbf{c}$  has weight  $\neq t')$  **then**

**return erasure.**

**else**

**return m.**

**4.1 Security Analysis**

Next, we show that Algorithm 4.2 is an instance of a  $\gamma$ -Gap Rabin OT according to Definition 2.2.

**Correctness.** Observe that if parameters are chosen carefully and every party follows the protocol, Algorithm 4.2 works correctly. Let us assume that  $l = n$  and  $t' = 2t + 1$ . Let  $(\mathbf{e}_1, \mathbf{e}_2) = \mathbf{e}(\mathbf{P}')^{-1} = \mathbf{c}(\mathbf{P}')^{-1} + \mathbf{m} [\mathbf{G}^{\text{pub}}|\mathbf{G}']$ , where  $\mathbf{e}_1$  is an  $n$ -bit vector. Then, iff  $\mathbf{e}_1$  has weight  $\leq t$ , the decryption procedure returns the correct message  $\mathbf{m}$ . Else an erasure occurs or the receiver obtains a false message  $\mathbf{m}' \neq \mathbf{m}$ . However, the latter case is unlikely to appear, since then, the weight of  $\mathbf{m}'\mathbf{O}^{\text{pub}} + \mathbf{c}$  is  $t'$ . Thus,  $\mathbf{m}'\mathbf{O}^{\text{pub}} + \mathbf{c} + \mathbf{m}\mathbf{O}^{\text{pub}} + \mathbf{c}$  has weight  $\leq 2t' = 4t + 2$ . It is easy to check that, for the reasonable parameters ( $m > 10$  and appropriate  $t$ ), the codeword  $(\mathbf{m} + \mathbf{m}')\mathbf{O}^{\text{pub}}$  has weight below the Gilbert-Varshamov (GV) bound for  $\mathbf{O}^{\text{pub}}$ , which is infeasible to find, even if such a codeword exists.

Since every choice of  $t'$  below half of the GV-bound of  $\mathbf{O}^{\text{pub}}$  leads to a correct scheme, the parameters may be chosen, such that the probability  $\mathcal{P}$  of obtaining the message  $\mathbf{m}$  varies. We can compute  $\mathcal{P}$  as the fraction of error vectors with no more than  $t$  entries on the positions of  $\mathbf{G}^{\text{pub}}$ :

$$\mathcal{P} := \sum_{i=0}^t \frac{\binom{n}{i} \binom{l}{t'-i}}{\binom{n+l}{t'}} = 1 - \underbrace{\sum_{i=t+1}^{t'} \frac{\binom{n}{i} \binom{l}{t'-i}}{\binom{n+l}{t'}}}_{=: \mathcal{Q}^{\text{H}}}. \tag{1}$$

Thus, for instance, if  $n = l$  and  $t' = 2t + 1$ , GV bound for  $\mathbf{O}^{\text{pub}}$ , the scheme works correctly and we have  $\mathcal{P} = \mathcal{Q}^{\text{H}} = 1/2$ .

**Gap.** In fact, Algorithm 4.2 implements the  $\gamma$ -Gap Rabin OT for some  $\gamma > 0$ , since even an honest-but-curious receiver has the possibility to raise its probability of receiving the message  $\mathbf{m}$ . He might choose to guess a part of the error vector or try to apply a general decoding algorithm to the erroneous word  $\mathbf{c}_2$  of the code  $\mathbf{G}'$ . These attacks are reflected in the following formula for the probability  $\mathcal{Q}_A$  of an erasure for a dishonest receiver spending  $A$  operations on decryption:

$$\mathcal{Q}_A = \sum_{i=t_0}^{t_1} \frac{\binom{n}{i} \binom{l}{t'-i}}{\binom{n+l}{t'}}$$

where  $t_0 > t + 1$ ,  $t_1 < t'$  (compare (1)) and the following conditions hold:

- (i) Solving General Syndrome Decoding problem for  $\mathbf{c}$  and  $\mathbf{O}^{\text{pub}}$  takes more than  $A$  operations. Note that  $A$  can be computed taking into account the (best known) attack by Canteaut and Chabaud [5] using the lower bound from [24]:

$$A \geq 2^{-t'} \log_2(1-k/(n+l)). \quad (2)$$

- (ii) General Syndrome Decoding problem for  $\mathbf{c}_2 = \mathbf{m}\mathbf{G}' + \mathbf{e}_2$  and  $\mathbf{G}'$  cannot be solved in  $A$  operations if  $\mathbf{e}_2$  has weight  $\geq t' - t_1$ .
- (iii) If the weight  $w$  of  $\mathbf{e}_1 = \mathbf{c}_1 + \mathbf{m}\mathbf{G}^{\text{pub}}$  is larger than  $t_0$ , the receiver cannot guess a sufficiently large subset of the support of  $\mathbf{e}_1$  to apply the decoding algorithm for Goppa codes. This is

$$\frac{\binom{n}{w-t}}{\binom{w}{w-t}} m^3 t^2 \geq A, \quad (3)$$

since each decoding attempt takes  $m^3 t^2$  operations [6] and there are  $\binom{w}{w-t}$  correct guesses.

To the best of our knowledge there exist neither codes with better error correction ratio than binary irreducible Goppa codes nor efficient list decoding algorithm for binary irreducible Goppa codes [13]. Thus, if  $\text{wt}(\mathbf{e}_1) > t_0$ , the receiver either has to guess part of the error or is forced to use a general decoding algorithm.

We conclude that the dishonest receiver can achieve  $\mathcal{Q} = 1$  only if general decoding is easy. The gap is computed as  $\gamma = \mathcal{Q}^H - \mathcal{Q}_A$ . Given  $A$ , the parameters of Algorithm 4.2 must be chosen according to (2). Condition (ii) allows us to compute  $t_1$  by substituting in (2):  $t'$  and  $l + n$  with  $t' - t_1$  and  $n$ , respectively. Finally,  $t_0$  is equal to minimal  $w$ , satisfying (3). In Appendix A.1, we present the numerical computations of  $\mathcal{Q}^H$  and  $\mathcal{Q}_A$  for  $A = \{38, 70\}$  and some proposed parameter sets.

**Sender's Security.** It appeared to be difficult to show a reduction to a hard problem. The general intuition behind the sender's security of our scheme is as follows. Assume that there exists a malicious receiver algorithm  $R$  which can recover all the messages in the case of erasure. Then,  $R$  must efficiently perform either of the following tasks: 1) Correct substantially more than  $t$  errors in a  $(n, k \geq n - mt)$  Goppa code; 2) Extend a Goppa code in such a way that the

extended code efficiently corrects as many errors as the Goppa code of the same size.

As there is no known polynomial algorithm for either of these problems, we believe that the sender's security can be achieved in principle.

### Receiver's Security

**Proof** assures that  $\mathbf{e}$  is of weight  $t'$ . In other words, the dishonest sender cannot influence  $\mathcal{P}$  by playing with the error vector's weight. His ability to do it would contradict to security of Stern's ZKID.

*Remark 4.1.* We note that Stern's ZKID can be used for proving the validity of the McEliece encryption in the same way as it is used for the OT transmission in Algorithm 4.2. This yields a verifiable variant of McEliece encryption. Verifiable encryption has numerous applications in cryptographic protocol theory (see [4]). We leave a formal treatment of this subject for the separate paper.

**Theorem 4.2.** *The Sender  $S$  who can distinguish  $\text{Views}_S(\mathbf{m}|E = 0)$  and  $\text{Views}_S(\mathbf{m}|E = 1)$  can distinguish a Goppa code from a random matrix.*

*Proof (Sketch)*

The proof goes in two parts: First, we show that a sender  $S$  able to efficiently detect erasures (with probability 1) can distinguish the Goppa part of  $\mathcal{O}^{\text{pub}}$  from the random part. Second, we build an oracle (in a straight forward way) which distinguishes a Goppa code from a random code using  $S$ . For simplicity of our presentation, we take  $l = n$  and  $t' = 2t + 1$ , however our proof generalizes to all other parameter sets.<sup>1</sup>

Note, that in the case of erasure, the probability  $p_i$  that for a position  $i$  of  $\mathcal{G}^{\text{pub}}$ ,  $\mathbf{e}_i = 1$  is at least  $(t + 1)/n$ , while for a position  $j$  of  $\mathcal{G}'$  the probability  $p_j$  that  $\mathbf{e}_j = 1$  is at most  $t/l$ . By heuristics, we can construct a distinguisher  $D$  which can tell the positions of  $\mathcal{G}^{\text{pub}}$  apart from the ones in  $\mathcal{G}'$ . This takes  $\mathcal{O}(\text{running-time}(S) \cdot n^2)$  steps since  $|p_i - p_j| \geq 1/n$ . This approach only fails if  $p_i \approx p_j$ , which we can fix by guessing some positions of  $\mathcal{G}^{\text{pub}}$ . See Appendix A.3 for details.

Now, given two matrices  $\mathcal{G}_1$  and  $\mathcal{G}_2$ , where one is a Goppa code, we can tell, which one is the Goppa code, by querying  $D$  with  $\mathcal{O}^{\text{pub}} = [\mathcal{G}_1 \mid \mathcal{G}_2]$  for the Goppa part of  $\mathcal{O}^{\text{pub}}$ .  $\square$

## 4.2 Omitting Trusted Third Party

In a fully secure scheme, the key generation is performed by the receiver and its correctness is verified by the sender using Shamir's PKP ZKID [25] (see Section 2). The basic idea is that the receiver computes the public key  $\mathcal{O}^{\text{pub}}$ , while the sender provides its random part  $\mathcal{G}'$  key and checks correctness by Shamir's ZKID. The key generation protocol is summarized in Algorithm 4.3.

The last step of Algorithm 4.3 requires some additional explanation: After the second last step, the sender knows a  $k' \times n$  submatrix  $\mathbf{K}$  of  $[\mathcal{G}^{\text{pub}} \mid \mathcal{G}']$  and

<sup>1</sup> Here we refer to the parameter sets which provide for secure McEliece encryption.

---

**Algorithm 4.3.** Public Key Generation Without TTP

---

**Input:** Security parameters same as in Algorithm 4.1 and  $k' < 2^m - mt \in \mathbb{N}$ .**Output:** The public key  $(\mathbf{O}^{\text{pub}}, t')$ .**Receiver:** Generate the same public key  $(\mathbf{G}^{\text{pub}}, t)$  as in Algorithm 4.1.Choose a  $(n + l) \times (n + l)$  random permutation matrix  $\mathbf{P}'$ .Commit to the rows of  $\mathbf{G}^{\text{pub}}$  (one by one,  $k$  commitments in total).**Sender:** Generate a random matrix  $\mathbf{G}' \in \mathbb{F}_2^{k \times l}$  and send it to the receiver.**Receiver:** Publish the systematic matrix  $\mathbf{O}^{\text{pub}}$  generating the same  $[n + l, k]$  code as  $[\mathbf{G}^{\text{pub}} | \mathbf{G}'] \mathbf{P}'$ .**Sender:** Choose a random subset  $\mathcal{K}'$  of cardinality  $k'$  from  $\{1, \dots, k\}$ .Ask the receiver to reveal the commitments for  $\mathcal{K}'$ .Compute the rows of  $[\mathbf{G}^{\text{pub}} | \mathbf{G}']$  with indices in  $\mathcal{K}'$ .**Receiver:** Use Shamir's ZKID to prove to the sender, that there is a permutation, such that the rows of  $[\mathbf{G}^{\text{pub}} | \mathbf{G}']$  with indices in  $\mathcal{K}'$  are simultaneously in the code generated by  $\mathbf{O}^{\text{pub}}$ .

---

can compute the  $n + l - k$  dimensional kernel of  $\mathbf{O}^{\text{pub}}$  given by a matrix  $\mathbf{H}$ . Now we can take  $\mathbf{H}$  and  $k'$  as system parameters for Shamir's Permuted Kernel ZKID. If the receiver is honest and has followed the protocol, he knows the secret permutation  $\Pi = \mathbf{P}'$  corresponding to the user's identity  $\mathbf{K}$ , i.e. a permutation such that  $\mathbf{K} \cdot \Pi \cdot \mathbf{H} = \mathbf{0}$ . Thus, the honest receiver can employ Shamir's ZKID to convince the sender by a zero-knowledge proof that he followed the protocol, while the dishonest receiver will be revealed.

Note that  $\mathbf{G}'$  can be generated using a pseudorandom generator. In this case, only a seed to the generator needs to be transmitted, hereby communication cost can be reduced.

## 5 1-out-of-2 String Oblivious Transfer

As a generalization of the previous scheme, we can construct a substantially more efficient protocol for 1-out-of-2 String OT. Unfortunately, this construction inherits the drawback of the previous scheme – we are unable to formally prove its sender's security. In fact, the sender's security proof would be a generalization of that of our Rabin OT scheme.

In this section, we briefly sketch this 1-out-of-2 String OT scheme and its security intuition, without giving formal proofs.

We assume the players' inputs to the protocol to be random since there exists a reduction by Beaver [2] which allows to convert such randomized OT into OT with actual inputs. Let the following be the security parameters: the matrix  $\mathbf{Q} \in \mathbb{F}_2^{k \times n}$  chosen uniformly at random, and  $\mathbf{G} \in \mathbb{F}_2^{k \times n}$ , as before, be a canonical generator matrix of an  $[n, k \geq n - mt, 2t + 1]$  binary irreducible Goppa code. Encryption is done using a variant of the McEliece PKC, we assume that the corresponding inputs' length is whatever prescribed by the encryption algorithm, denoted  $a$  bits for certainty.

Our 1-out-of-2 String OT protocols is summarized in Algorithm 5.1.

---

**Algorithm 5.1.** 1-out-of-2 String OT

---

**Input: Security parameters:**  $m, k, k', t, G, Q$ **Sender:**  $\mathbf{b}_0, \mathbf{b}_1 \in \mathbb{F}_2^m$ ; **Receiver:**  $c \in \{0, 1\}$ **Output: Sender:** none; **Receiver:**  $b_c$ **Receiver:** Generate random permutation matrices  $P', P'' \in \mathbb{F}_2^{n \times n}$  and a random matrix of rank  $k'$ :  $S' \in \mathbb{F}_2^{k' \times k}$ . Set  $C_c = S'GP'$ ,  $C_{1-c} = S'QP''$ .Send  $[C_0|C_1]$  to the receiver.Prove using Shamir's ZKID that  $[C_0|C_1]$  is a permuted subcode of  $[G|Q]$ .**Sender:** Reject if the proof fails, otherwiseFor  $i = 0, 1$ : Encrypt  $\mathbf{b}_i$  as follows:  $\mathbf{b}_i C_i + \mathbf{e}_i$ , where  $\mathbf{e}_i$  is random vector of weight  $t$ , and send the encryption.**Receiver:** Decrypt and output  $b_c$ .

---

Note that the communication cost of this protocol can be substantially reduced, if only the non-systematic parts of the codes are dealt with. This modification will also require using the IND-CCA2 conversion for encryption [17]. Also,  $Q$  can be computed using a pseudorandom generator such that only its seed will be obtained by coin flipping.

**Intuition.** We provide only a sketch of the security analysis.

**Correctness.** If both players follow the protocol, then the receiver is not rejected by the sender and is able to decode  $C_c$  which is a subcode of  $G$ . Hence, the decoding algorithm of  $G$  can be used.

**Sender's Security.** Assume that the receiver is honest-but-curious, i.e., he tries to compute both inputs, but still follows the protocol. In order to compute  $\mathbf{b}_{1-c}$ , he must decode a subcode of the random code, i.e., solve Problem (ii) of Definition 2.3.

Now, suppose that the IND-CPA version of the McEliece PKC [21] is used for encryption, then the input  $\mathbf{b}_{1-c}$  which is encrypted on the subcode of  $Q$  is indistinguishable from random according [21, Lemma 3].

If we use only non-systematic parts of the codes, then we must employ the IND-CCA2 conversion [17] since the "message" part of the encryption will be sent in open in this case. Indistinguishability of one of the inputs will follow in a way similar to the variant above. Note, that the price to pay here is the additional assumption: the random oracle model, which is not required by the variant above.

Now, assume that the receiver is fully malicious. The ZK proof will convince the sender that  $[C_c|C_{1-c}]$  is indeed a permutation of  $[G|Q]$ . Unfortunately, the receiver may distribute the columns of  $G$  and  $Q$  into both  $C_c$  and  $C_{1-c}$ . Then, the security proof will boil down to proving the receiver's inability to efficiently decode an extended Goppa code. In fact, we will need a generalization of the security proof for our Rabin OT protocol: here, the receiver does not necessarily distributes  $G$  and  $Q$  as "half-to-half" in the subcodes. As it was mentioned in the previous section, such the proof is not easy to construct.

**Receiver's Security.** The malicious sender who learns the receiver's choice must either distinguish the subcode of  $G$  from that of  $Q$  hence solving Problem (i) or recover the permutations  $P', P''$  solving Problem (iv) of Definition 2.3.

**Employing Cut-And-Choose.** We note, although do not prove it formally, that in the above algorithm, one can use the cut-and-choose technique instead of the zero-knowledge proof in order for the sender to check that the keys were formed correctly. This would remove the above mentioned problem with the sender's security proof. However, in this case, our protocol becomes quite similar to that of [9], since it would use the same machinery to reduce the advantage of malicious receiver. Therefore, we omit details due to space limitations.

**Acknowledgments.** The authors would like to thank anonymous reviewers from Eurocrypt 2008 for their comments. The second author would also like to thank Yang Cui for his pointer on verifiable encryption.

## References

1. Aiello, W., Ishai, Y., Reingold, O.: Priced oblivious transfer: How to sell digital goods. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 119–135. Springer, Heidelberg (2001)
2. Beaver, D.: Precomputing oblivious transfer. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 97–109. Springer, Heidelberg (1995)
3. Bellare, M., Micali, S.: Non-interactive oblivious transfer and applications. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 547–557. Springer, Heidelberg (1990)
4. Camenisch, J., Shoup, V.: Practical verifiable encryption and decryption of discrete logarithms. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 126–144. Springer, Heidelberg (2003)
5. Canteaut, A., Chabaud, F.: A new algorithm for finding minimum-weight words in a linear code: Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. IEEEITIT: IEEE Transactions on Information Theory 44 (1998)
6. Courtois, N., Finiasz, M., Sendrier, N.: How to achieve a McEliece-based digital signature scheme. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 157–174. Springer, Heidelberg (2001)
7. Crépeau, C.: Equivalence between two flavours of oblivious transfers. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 350–354. Springer, Heidelberg (1988)
8. Damgård, I., Nielsen, J.: Commitment schemes and zero-knowledge protocols. Lecture notes, University of Aarhus (February 2008), <http://www.daimi.au.dk/~ivan/ComZK08.pdf>
9. Dowsley, R., van de Graaf, J., Müller-Quade, J., Nascimento, A.: Oblivious transfer based on the mcEliece assumptions. In: Safavi-Naini, R. (ed.) ICITS 2008. LNCS, vol. 5155, pp. 107–117. Springer, Heidelberg (2008)
10. Even, S., Goldreich, O., Lempel, A.: A randomized protocol for signing contracts. Commun. ACM 28(6), 637–647 (1985)
11. Goldreich, O.: Foundations of Cryptography - Volume 2 (Basic Applications). Cambridge University Press, Cambridge (2004)

12. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: STOC, pp. 218–229. ACM, New York (1987)
13. Guruswami, V., Sudan, M.: Improved decoding of reed-solomon and algebraic-geometry codes. *IEEE Transactions on Information Theory* 45(6), 1757–1767 (1999)
14. Haitner, I.: Implementing oblivious transfer using collection of dense trapdoor permutations. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 394–409. Springer, Heidelberg (2004)
15. Kalai, Y.: Smooth projective hashing and two-message oblivious transfer. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 78–95. Springer, Heidelberg (2005)
16. Kilian, J.: Founding cryptography on oblivious transfer. In: STOC, pp. 20–31. ACM, New York (1988)
17. Kobara, K., Imai, H.: Semantically secure McEliece public-key cryptosystems - conversions for McEliece PKC. In: Kim, K.-c. (ed.) PKC 2001. LNCS, vol. 1992. Springer, Heidelberg (2001)
18. MacWilliams, F.J., Sloane, N.J.A.: *The Theory of Error-Correction Codes*, 7th edn. North-Holland, Amsterdam (1992)
19. McEliece, R.J.: A public key cryptosystem based on algebraic coding theory. DSN progress report, 42–44, 114–116 (1978)
20. Naor, M., Pinkas, B.: Efficient oblivious transfer protocols. In: SODA, pp. 448–457 (2001)
21. Nojima, R., Imai, H., Kobara, K., Morozov, K.: Semantic security for the McEliece cryptosystem without random oracles. In: Charpin, P., Helleseht, T. (eds.) *Designs, Codes and Cryptography*, vol. 49(1-3), pp. 289–305. Springer, Heidelberg (2008)
22. Poupard, G.: A realistic security analysis of identification schemes based on combinatorial problems. *European Transactions on Telecommunications* 8(5), 417–480 (1997)
23. Rabin, M.O.: How to exchange secrets by oblivious transfer. Technical report, Aiken Computation Laboratory, Harvard University, Tech. Memo TR-81 (1981)
24. Sendrier, N.: On the security of the McEliece public-key cryptosystem. In: Blaum, M., Farrell, P.G., van Tilborg, H. (eds.) *Proceedings of Workshop honoring Prof. Bob McEliece on his 60th birthday*, pp. 141–163. Kluwer, Dordrecht (2002)
25. Shamir, A.: An efficient identification scheme based on permuted kernels. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 606–609. Springer, Heidelberg (1990)
26. Stern, J.: A new identification scheme based on syndrome decoding. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 13–21. Springer, Heidelberg (1994)
27. Vaudenay, S.: Cryptanalysis of the Chor–Rivest cryptosystem. *J. Cryptology* 14(2), 87–100 (2001)
28. Wiesner, S.: Conjugate coding. *SIGACT News* 15(1), 78–88 (1983)

## Appendix A: Details on Security of Rabin OT

### A.1 Examples of Security Parameters

We can achieve reasonable values for  $\mathcal{P}$  and  $\mathcal{Q}_{280}$ , compare Table A.1. In fact, one can even use the dishonest receiver’s strategy in a positive way, i.e., to raise

**Table A.1.** Parameter sets for the Rabin OT

Parameters				Size Public Key	Size Ciphertext	Decryption	$\mathcal{Q}_A$		
$m$	$t$	$t'$	$l$	$= k(n + l - k)$	$= n + l - k$	runtime	$\mathcal{Q}^H$	$A = 2^{38}$	$A = 2^{80}$
12	200	$2t + 1$	$2^{12}$	1,377 KBytes	812 Bytes	$2^{26}$	0.5	0.41	0.11
13	402	$2t + 13$	$2^{13}$	4,974 KBytes	1,677 Bytes	$2^{28.5}$	0.66	0.61	0.36
14	800	$2t + 1$	$2^{14}$	17,874 KBytes	3,448 Bytes	$2^{31}$	0.5	0.46	0.29

the chance of obtaining the message. The work factor for decryption is then given by Equation (3). This is useful for protocols like in [7], where we need to ensure that the receiver gets at least half of the messages, compare Section 3.

*Example 1.* With the first parameter set from Table A.1 and a receiver spending up to  $2^{35}$  operations on each decryption, we can obtain a 1-out-of-2 OT by Crépeau’s construction, which fails with probability less than  $2^{-30}$  if we choose  $N = 180$ .

## A.2 Reaction Attack

Note that if the same public key  $\mathcal{O}^{\text{pub}}$  is used in the different instances of Algorithm 4.2, the dishonest sender can adaptively influence the erasure probability, as long as he gets feedback whether an erasure occurred or not. Suppose that an attacker learns that the receiver cannot decode a certain ciphertext. Then, the sender can choose to modify the corresponding error vector only slightly for the next encryption. Thus, by statistics, the sender could identify the columns of  $\mathcal{G}^{\text{pub}}$  in  $\mathcal{O}^{\text{pub}}$ , which breaks the security of our scheme. Nevertheless, the sender should be cautious, as the receiver might detect such manipulations by comparing ciphertexts.

This might get important as feedback may well come from the higher level protocols (like Crépeau’s protocol), for which oblivious transfer is used as a primitive. However, there are plenty of possible countermeasures against an attack by feedback. For instance, when the conversion [17] is used for encryption, the task of tuning the erasure probability is not at all trivial.

## A.3 Proof Details for Receiver’s Security

The distinguisher  $D$  works as follows. Repeat the following until  $n^{2+\epsilon}$  ( $0 < \epsilon < 1$ ) erasure views are encountered:

- Generate a view of the sender using some error vector  $\mathbf{e}$  (distinct each time) and submit it to  $S$
- Each time  $S$  outputs “erasure”, remember on which columns the error locations of  $\mathbf{e}$  were.

Note that the expected running time will be  $n^{2+\epsilon}/\mathcal{Q}$ , where  $\mathcal{Q}$  is the erasure probability. Hence,  $D$  is efficient.

Then, consider the “score” of each column. For those of  $\mathbf{G}^{\text{pub}}$ , the expected score is at least  $(t+1)n^{1+\epsilon}$ , since at least  $t+1$  errors are needed to cause an erasure. Hence, for the columns of  $\mathbf{G}'$ , the expected score is at most  $(t'-t-1)n^{2+\epsilon}/l$  as at most  $t'-t-1$  errors are left for  $\mathbf{G}'$ . Now, the standard Chernoff bound can be applied in order to show that one can distinguish between two random variables with the above expectations with negligible (in  $n$ ) probability of error.

It is easy to check that the above reasoning works when  $(t+1)/n > (t'-t-1)/l$ . The only case, when it breaks down is when the above expectations are too close to each other such that the Chernoff bound does not apply. However, this can be fixed in the following way: guess a few positions of  $\mathbf{G}^{\text{pub}}$  and run  $D$ . Note that once  $t+1$  candidate columns for  $\mathbf{G}^{\text{pub}}$  are obtained, they can be easily verified by placing the error locations on them and submitting such the view to  $S$ . If the initial guess was wrong, guess a different set of columns and start over. Since the probability of the correct guess is non-negligible, the expected running time is polynomial in  $n$ .

In case of  $(t'-t-1)/l > (t+1)/n$ , we slightly modify the construction of  $D$ : it will iterate until  $n^{2+\epsilon}$  *non-erasure* views are encountered. Note that in this case, the expected score of  $\mathbf{G}^{\text{pub}}$ 's column is at most  $tn^{1+\epsilon}$ , while that of  $\mathbf{G}'$  is at least  $(t'-t)n^{2+\epsilon}/l$ . Thus, the reasoning before applies in a similar way.

# Protection of Sensitive Security Parameters in Integrated Circuits

Dejan E. Lazich<sup>1</sup> and Micaela Wuensche<sup>2</sup>

<sup>1</sup> Micronas GmbH  
Hans-Bunte-Str. 19  
D-79108 Freiburg  
lazic@ira.uka.de

<sup>2</sup> Institut fuer Algorithmen und Kognitive Systeme (IAKS)  
University of Karlsruhe  
Am Fasanengarten 5  
D-76131 Karlsruhe

**Abstract.** To protect sensitive security parameters in the non-volatile memory of integrated circuits, a device is designed that generates a special secret key (called IC-Eigenkey) to symmetrically encrypt this data. The IC-Eigenkey is generated by the integrated circuit itself and therefore unknown to anybody else. The desired properties of such an IC-Eigenkey are postulated and a theoretical limit on the distribution of IC-Eigenkeys over an IC-production series is derived. The design of the IC-Eigenkey generator is based on silicon physical uncloneable functions. It exploits the marginal random variations of the propagation delays of gates and wires in an integrated circuit. A method is introduced that uses codewords of error control codes to configure the IC-Eigenkey generator in a way that the generated bits are as statistically independent of each other as possible.

**Keywords:** Cryptography, Coding Theory, Signal Processing, Cryptographic Module, Sensitive Security Parameters, Physical Uncloneable Functions, Error Control Codes.

## 1 Introduction

*Monolithic integrated circuits* (ICs), well known as *chips*, are basic hardware components of present-day *cryptographic modules* that are designed to provide security services in communication networks. In order to be protected against all kinds of physical attacks, the relevant security parts of a cryptographic module are placed inside of its *cryptographic boundary*, including all critical hardware and software components. The size of a cryptographic boundary can range from a chipset, a group of ICs designed to work together, to a small area on a single chip. It covers one or more *cryptographic devices* that execute specified cryptographic functions. Every time a cryptographic device has to carry out a cryptographic function, all sensitive data needed for this task must be transferred to this device

before the execution of the function. This data itself has to be protected against unauthorized disclosure and modification.

An IC, designed to carry out cryptographic functions, can use a variety of secret, private and public cryptographic keys, authentication data such as passwords and PINs, as well as other kinds of sensitive data whose disclosure and/or modification can compromise the security of its cryptographic function. Throughout this paper, all such critical data an IC needs to complete its designated security functions will be called *Sensitive Security Parameters* (SSPs) [1]. Some SSPs will be transferred to the IC during the personalization phase of its lifecycle; others can be generated during its operational deployment in the end-user phase, either on its own or by communicating with other ICs (in the same or in another cryptographic module) using different key transport/agreement protocols. As soon as SSPs arise, some of them must be stored in non-volatile memory, to be used later for certain security tasks whenever they are needed during the IC's operational deployment.

During the whole lifecycle of an IC the stored SSPs should be protected against all relevant kinds of *implementation attacks* [2]. These attacks represent all unauthorized manipulations (tampering) that take advantage of inherent flaws and weaknesses in the design and physical implementation of cryptographic functions in an IC. The usual approach to effective protection consists in several kinds of *tamper-resistant* countermeasures which can be combined with *tamper-responsive* actions and *tamper-evident* indications. Currently, four main types of *tamper-resistant* countermeasures, called *security mechanisms*, are in use [3]. The *Physical security mechanism* prevents an attacker from accessing the processing, connecting and storage elements of the IC. The *Data storage cell security mechanism* refers to the resistance that the IC presents to active and passive probing of storage cells. It is assumed that attacks against the data storage cell security mechanism aim only at measuring the charge of the storage cell and not at gaining access to the circuitry, which is a property of the physical security mechanism. The IC's *Environmental exchange security mechanism* is designed to protect the IC against changes in its environment that will take the IC out of its normal operating mode. The object of the *Leakage security mechanism* is to consider the emanations from the IC under a varying set of environmental conditions.

All these security mechanisms are mainly considered against *dynamic attacks*, when the IC is operating and just executes its cryptographic function. However, in case of the data storage cell security mechanism for protection of plaintext SSPs in non-volatile memory, *static attacks* on this non-volatile memory have to be taken into consideration, too. Attacks of this kind can take place at any time, even if the IC is not currently executing its cryptographic function, but in standby or even disconnected from power. Protecting an IC against a static attack on its non-volatile memory where SSPs are stored in plaintext is especially difficult, since in this case the attacker often has enough time and is undisturbed while attacking cryptographic modules embedded in risky environments. In contrast to this, a dynamical attack is subject to constraints regarding the

timing, the short duration and the protected flow of the crypto algorithms that are executed.

Aside from the data storage cell security mechanism, a different approach exists to protect SSPs in non-volatile memory, so that in this case only measures against dynamic implementation attacks are necessary. In this approach, the SSPs are symmetrically encrypted using a special secret key, which is generated by the IC itself and therefore unknown to anybody else. In their encrypted form, the SSPs can be stored in any non-volatile memory in or outside the cryptographic boundary, or even outside the cryptographic module. In this case, the data storage cell security mechanism is no longer necessary to protect the non-volatile memory containing the SSPs in plaintext. The special secret key used for the encryption/decryption of the SSPs must be a binary, time-invariant true random number which will be called the *IC-EigenKey* (IC-EK) throughout the paper.

Due to the vital importance of the IC-EK to the security of the whole cryptographic module, standard methods for generation and non-volatile storage of random cryptographic keys in an IC [4] are not sufficient in this case. Particular physical properties and technical mechanisms must be used to keep the complete information about the IC-EK in a hidden analog form within the IC during its complete lifecycle. Only on demand (when some SSPs must be de- or encrypted) and only by means of a suitable *extraction circuit* can the IC-EK be extracted from this disguised, unrecognizable non-digital form and converted to its binary form. The binary form of the IC-EK should only be applied as short as possible and afterwards be deleted instantly and without a trace. During this short period the IC-EK is protected by all needed security mechanisms against dynamical attacks. Only its basic disguised analog form may be kept in the IC as necessary information for the next extraction of the binary form of the IC-EK.

Compared to the data storage cell security mechanism for protection of plaintext SSPs stored in non-volatile memory, the alternative with an IC-Eigenkey needs some additional data-processing, i.e. the extraction of the IC-EK to its digital form, the generation of a cryptographic hash value to check the IC-EK integrity and finally the decryption of the SSPs. On the other hand, the application of extensive and costly tamper resistance techniques for protecting the non-volatile memory is obsolete. This applies especially to multi-functional single-chip cryptographic modules in embedded systems. Such modules, known as *Systems on Chip* (SoC), are designed to carry out one or more non-cryptographic functions as a primary task, while additional cryptographic functions are realized on the same chip. SoCs are often embedded in risky environments and thus are more frequently subject to implementation attacks. They are also highly sensitive to growing manufacturing costs. In order to minimize these costs in SoCs, the cryptographic boundary is often reduced only to a small area on the chip with no additional chip-external protection measures. In this case, the protection of SSPs by IC-Eigenkeys can have some advantages compared to the data storage cell security mechanism. Therefore, in the present paper a device for the generation of a suitable IC-Eigenkey will be described.

The remainder of this paper is organized as follows. In Chapter 2, the desired properties of an IC-EK will be postulated and a theoretical limit on the distribution of IC-EKs over an IC-production series will be derived. In Chapter 3, different existing methods and technologies useful for the realization of an *IC-EK generator* will be presented, particularly a silicon-based alternative that is of special interest for the present paper. Chapter 4 introduces the architecture of an IC-EK generator. The configuration and control of this IC-EK generator will be discussed in Chapter 5, and simulation and test results for different configurations of IC-EK generators are described in Chapter 6.

## 2 Properties of the IC-Eigenkey

Every single IC (*IC-instance*) from an *IC-production series* of  $M$  identical ICs, manufactured using the same lithographic masks, has to generate a binary number of  $N$  bit length upon a call of an IC-EK *initialization command*. This number, which is the IC-Eigenkey (IC-EK), is unique for this one IC-instance and unpredictable before it has been generated for the first time. Subsequently, every call of the IC-EK initialization command in this IC-instance must generate the same number again, regardless if the IC has been in standby or even disconnected from power in between. The IC-Eigenkey of an IC-instance may only be used for the protection of sensitive security parameters and for the generation of authenticated identification-data (IC-ID) for an IC-instance.

The IC-EK has to meet the following requirements:

1. *Privacy* - At no point during the whole lifecycle of the IC may the IC-EK be disclosed to anybody, nor be reproduced at an unintended location.
2. *Integrity* - The value of the IC-EK may not be susceptible to influences via the IC-ports nor in any other way and thus be subject to change. It may solely be generated by the *IC-Eigenkey generator* located inside of the IC's cryptographic boundary.
3. *Local and temporal restrictions* - The IC-EK may never leave the IC's cryptographic boundary, nor be stored in a non-volatile memory. It may only be available to the circuits for the symmetric de- and encryption of the SSPs. The volatile storage of the IC-EK, if necessary, may only be effected for a short time and preferably bitwise. After usage in the de- or encryption process of SSPs, its digital value must be deleted instantly. After decryption, the SSPs will be made available in plaintext to their respective users (that execute the envisaged crypto-algorithms inside the cryptographic boundary) and right after their application they must be deleted at once.
4. *Resistance to dynamical implementation attacks* - The IC's cryptographic boundary, and within it the IC-Eigenkey generator itself, must be protected against all kinds of dynamical implementation attacks.
5. *Avoidance of obfuscation* - In order to avert attacks by reverse engineering, the IC-EK generator may not operate based on a hidden deterministic algorithm or on a known deterministic algorithm with hidden parameters.

6. *Nondeterministic generation* - The operation of the IC-EK generator must be based on a random analog physical process. Each bit value of the IC-EK must be derived from a true random signal obtained from this physical process.
7. *Time invariance* - The set of IC-EK generators of an IC-production series should be regarded as a unique true random number generator which every time generates the same binary number in the same IC-instance, while these numbers are randomly distributed over the instances of the IC-production series.
8. *Non-cloneability* - The correct bit value of the IC-EK can be derived from the true random signal only by an *extraction circuit*, which is inseparably integrated in the IC-EK generator, and only after an IC-EK initialization command has been called. This binary value of the IC-EK may by no other means and methods be reconstructable.
9. *Reliability* - The probability of a misinterpretation during the derivation of IC-EK bits by the extraction circuit, caused by various environmental disturbances and measurement errors, should be as small as possible. This IC-EK *Bit Extraction Error Probability* (BEEP) can be additionally reduced by using error correcting codes, if the BEEP value is not too high and therefore out of the channel capacity region. To provide an integrity check, a cryptographic hash value of the correct IC-EK can be computed during the personalization phase of the IC's lifecycle and kept in the IC's non-volatile memory as plaintext.
10. *Uniform distribution* - The smallest Hamming distance between any two IC-EKs generated in different IC-instances of an IC-production series of identical ICs should be as large as possible. This is necessary in order to keep an exhaustive search for further IC-EKs as comprehensive as possible, after one or more IC-EKs have been discovered (compromised). Furthermore, the bit length  $N$  of the IC-EK should be long enough to ensure that the *maximal possible minimum Hamming distance* between IC-EKs,  $d_{H_m}$ , can reach an acceptance limit.

From an Information theory point of view, the  $N$ -bit IC-EKs of  $M$  different IC-instances in an IC-production series which are resistant against the exhaustive search attack should constitute the codewords of a *minimax binary block code*  $(N, M, d_{H_m})_{q=2}$  with a small code rate  $R = (\log_2 M)/N$ . For this case it was shown that the codewords are uniformly distributed over the complete encoding space containing  $2^N$  possible  $N$ -tuples [5]. Furthermore, it can be shown that random binary block codes, whose codewords represent random vectors with statistically independent and uniformly distributed binary random variables as components, asymptotically meet the properties of the minimax block codes [6]. Therefore, also the single random bits of the IC-EKs of an IC-production series should be mutually statistically independent, so that the IC-EKs, too, have a maximal possible minimum Hamming distance of each other, as is stated in requirement 10.

If the total number  $M$  of IC-instances with an IC-EK of given length  $N$  in an IC-production series is known, then from the *Gilbert-Varshamov bound* [7], [8] the maximal possible minimum Hamming distance  $d_{H_m}$  between two IC-EKs can be estimated. According to this asymptotic bound (when  $N \rightarrow \infty$ ) the highest code rate,  $R$ , of a binary block code with the given minimum distance  $d_{H_m}$  is lower-bounded by

$$R \leq 1 - H_2(\underline{d}_{H_m}) = 1 + \underline{d}_{H_m} \log_2(\underline{d}_{H_m}) + (1 - \underline{d}_{H_m}) \log_2(1 - \underline{d}_{H_m}) \quad , \quad (1)$$

where  $\underline{d}_{H_m} = d_{H_m}/N$  represents the normalized minimum distance and  $H_2(\cdot)$  the binary entropy function. For the here considered binary case,  $q = 2$ , the Gilbert-Varshamov bound is tight or almost tight (still unsolved problem), so that the derived estimations are highly accurate, if  $N > 100$  [9], [10], [11].

For example, if the IC-production series has a total of  $M = 2^{24}$  (ca. 17 millions) IC-instances with IC-EKs of length  $N = 256$ , then according to (1), the maximal possible minimum Hamming distance  $d_{H_m}$  will be larger than 80, if the IC-EKs are truly random and uniformly distributed over the Hamming space containing all possible  $2^{256}$  binary 256-tuples. With  $d_{H_m} \geq 80$  the security against the exhaustive search attack on IC-EKs in such an IC-production series would be high enough for most of the present-day security criteria. However, if the bits of an IC-EK are statistically dependent the minimum Hamming distance between two IC-EKs will be smaller and thus the resistance against the exhaustive search attack reduced. From this point of view, the main research objective is to develop an IC-EK generator which generates its IC-EK from bits whose mutual statistical dependencies are as small as possible.

### 3 Physical Uncloneable Functions

A promising approach to implement an IC-Eigenkey generator that would perform according to the above specifications uses so called *Physical Uncloneable Functions* (PUFs) [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], sometimes referred to as *Physical Random Functions* [12], [21], [22], or *Physical One-Way Functions* [23], [24].

These functions are implemented by hardware devices, that produce for any given input an output that uniquely identifies every single device (instance) of a series of identical hardware devices. The matching input/output pairs are called Challenge-Response Pairs (CRPs). To make sure that CRPs are in fact unique to each instance, the device is designed in a way that its response to a challenge depends on properties of the device that are subject to random process variations in manufacturing [25]. Due to the marginally different variations of these properties in the devices, each one will produce CRPs that are unique and specific to this instance. The random nature of process variations eliminates the possibility to control the responses of a device before it is manufactured and tested.

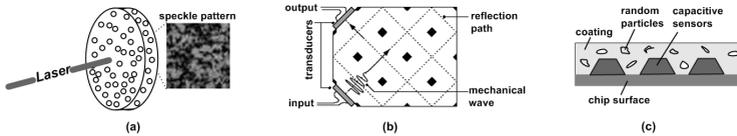
While many fundamentally different realizations have been described in literature, all authors agree on some basic properties and definitions which are

common to all PUFs [12], [13], [14], [15], [16], [17], [18], [19], [20]. They must be easy to evaluate, meaning they produce the response to a challenge within a relatively short time. A PUF is described as hard to characterize, if an attacker, using state-of-the-art technology, has no way of reconstructing it to a point where he could gain information about the response to a randomly chosen challenge. The commonly used term of manufacturer resistance refers to the property that it is technically impossible to produce two completely identical PUFs, because the process variations that are exploited to determine the PUF's response can not be influenced by today's manufacturing techniques. A PUF is said to be controlled, if the device that evaluates the response is physically connected to the PUF in an inseparable way. It controls which challenges are presented to the PUF and can hide its physical response, if only indirect information like encryption or hash shall be revealed to the outside world.

Several different approaches to realize a PUF have been described. They all differ in the basic type of the physical system whose manufacturing process variations are exploited to generate the PUF responses.

### 3.1 Different Basic Systems for PUFs

A representative example of an optical PUF is described in [23]. In this system, a laser beam is aimed through a transparent token and the resulting projection is processed into a digital response (see Fig. 1a). The token consists of optical-



**Fig. 1.** (a): Optical PUF [23]. (b): Acoustical PUF [20]. (c): Coating PUF [26].

grade epoxy, that has been mixed with micron-scale glass spheres. The random distribution of the spheres in the epoxy produces a characteristic speckle pattern in the projection. The speckle pattern is processed by the 2D Gabor Transform Algorithm. So the analog challenge, which consisted of the direction and wavelength of the laser beam, is transformed into a digital response.

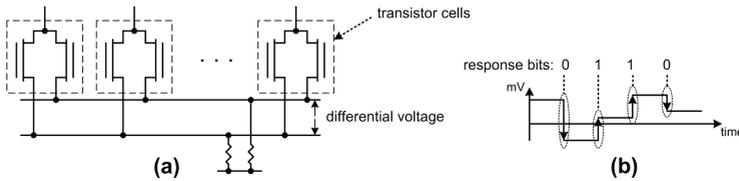
A coating PUF, as designed in [26], does not respond to different challenges, but generates only one fixed bit-sequence as value of the PUF. For this, an array of capacitive sensors is distributed over the surface of a chip and covered with a coating containing many randomly distributed particles with different dielectric constants (see Fig. 1c). Due to the inhomogeneity of the coating, information about the individual bits is derived from the local coating capacitances around the sensors.

In [20], an acoustical PUF based on a glass delay line is proposed. Digital signals are converted by a transducer into mechanical waves. These waves propagate through the glass of the delay line and are reflected several times at its

edges, until they reach an output transducer and are converted back into digital signals (see Fig. 1b). Inhomogeneities in the medium, as well as the conversion between analog and digital signals result in an attenuation and phase change of the output signal that are specific to each instance.

A type of PUF that has been described in several variations, is the so called *silicon PUF*. It is constructed only from semiconductor components, which can be affected by process variations in many different ways, each of which offers a different approach to PUF design.

On the lowest level of IC integration, a transistor-based PUF, called *integrated circuit identification device*, uses individually addressable MOSFET transistors to generate responses [27]. They are combined in cells of two transistors at a time, connected to load resistors. If a cell is activated, it produces a differential voltage across the load resistors that is directly dependent of the transistors randomly varying threshold voltages (see Fig. 2a). Stimulating the cells one by



**Fig. 2.** (a): Transistor cells of an integrated circuit identification device. (b): The resulting differential voltage.

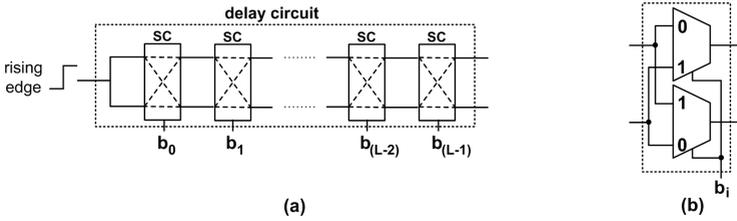
one, the random change of the differential voltage from one cell to the next will determine the PUF response, as depicted in Fig. 2b.

On a much higher level of IC integration complex circuits in an IC have been used as PUF. Response times of such a PUF to different challenges were converted to identification bits in order to identify single chips, as described in [28].

An efficient family of silicon PUFs uses propagation delay variations of gates and wires within an IC as the basic principle for PUF development. As this approach is especially suited as a basic pattern to design the IC-EK generator that is proposed in this paper, it will be explained in further detail in the next subsection.

### 3.2 Silicon PUFs Based on Propagation Delays of Gates

A family of silicon PUFs based on the propagation delay variations of gates and wires within the IC has been proposed in [12], [13], [14], [15], [18], [19]. These PUFs are mainly intended for IC identification and authentication purposes. Here, gates and wires are used as delay elements with specific but randomly distributed propagation delay variations. They are connected to form a delay circuit that consists of two separate delay lines, which can be simultaneously



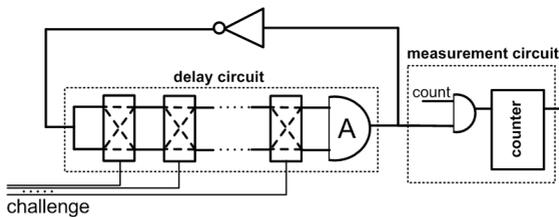
**Fig. 3.** (a): Delay circuit with two delay lines and  $L$  switching components SC. (b): Internal design of the switching components.

excited by a signal with a rising edge (see Fig. 3a). The delay lines lead through  $L$  *Switching Components* (SCs) that are capable of either passing on the signals in parallel to the following switching component, or crossing the delay lines, directing the signal on the upper line to the lower one, and vice versa. Each SC consists of two (2 to 1)-multiplexers, as depicted in Fig. 3b. So the actual path of each signal through the delay circuit and the resulting delay of each delay line is determined by an  $L$ -bit challenge vector, where each bit  $b_i$  ( $i = 0, \dots, (L - 1)$ ) corresponds to another switching component, setting it to either pass on or cross the signals. It is imperative that both delay lines be constructed perfectly symmetrical, so that differences in total line delays depend only on random propagation delay variations of their delay elements, not on a possibly asymmetrical layout.

To evaluate the delay variations and generate a PUF-response, two different designs have been introduced, each in several slightly different variations.

In an arbiter-based PUF version [18], [19], the measurement circuit that quantifies the response consists of a simple latch at the end of the delay circuit, that decides on which line the signal edge has traveled faster. An output bit is set accordingly. So in this design, each  $L$ -bit-challenge leads to a 1-bit-response.

Longer responses can be generated in a design based on a self-oscillating loop (see Fig. 4) [16], [13], [12]. This version uses the same delay lines, routed through the switching components that are controlled by an  $L$ -bit-challenge. Here, the racing edges will not be measured by an arbiter, but reduced to one single edge. This can be done by either connecting the outputs of both delay lines to the



**Fig. 4.** Silicon PUF based on a self-oscillating loop. The measurement circuit counts its oscillations and generates a response.

inputs of an *AND*-gate  $A$  (as in Fig. 4) [13], or by substituting the last switching component with a (2 to 1)-multiplexer so that the last challenge bit determines which edge will be chosen [12]. Either way, the remaining edge will be inverted and fed back into the beginning of the delay lines. Thus, a self-oscillating loop is created. The frequency of its oscillation is directly depending on the paths taken and on the random propagation delay variations of their gates and wires. The measurement circuit determines the frequency of the oscillation by counting the rising edges of the signal during a fixed period of time while the *count*-signal is high (see Fig. 4) [16]. The resulting counter-value will be given as response. So the length of the response to the  $L$ -bit-challenge depends on the oscillating frequency of the loop and the time interval of the measurement.

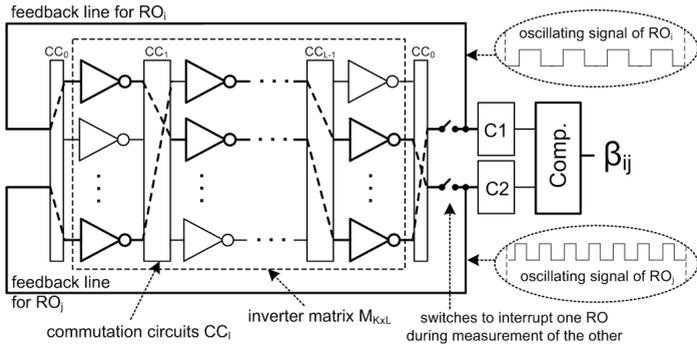
The arbiter-based PUF version as well as the one based on a self-oscillating loop both have advantages and disadvantages. The single self-oscillating loop is better suited to distinguish between different challenges, as the signals pass the delay line several times to accumulate the differences. However, the absolute measurement leaves it far more vulnerable to environmental changes, which can influence propagation delays of gates and wires to a great degree. In contrast to this, the arbiter version is less accurate in detecting the diminutive variations between the two delay lines, while it is far more resistant to environmental changes due to its relative measurement of the line delays which are influenced by the changes to the same extent.

The delay circuit described in this section consists of only two rows of  $L$  delay elements that can be entwined depending on the challenge vector. Because for each challenge the same pool of  $2L$  delay elements has to be used, and changes consist only in varying configurations, the resulting random bits will to a large degree be mutually statistically dependent.

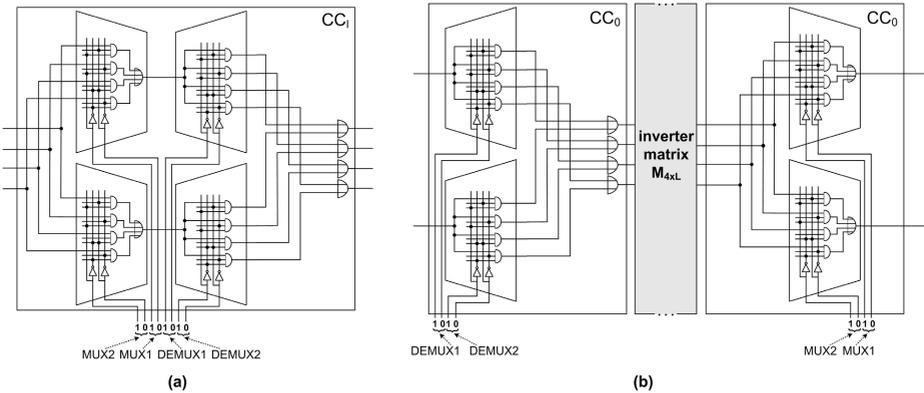
In the architecture presented in this paper and elaborated in Chapters 4 and 5, the reliability of the self-oscillating loop and the environmental stability of relatively measured delays will be combined in a single PUF circuit, while the statistical dependence of the generated bits will be reduced.

## 4 Delay Line Configuration of the IC-Eigenkey Generator

To implement an IC-EK generator as specified in Chapter 2 a new design of a silicon PUF based on the propagation delay variations of its gates and wires will be introduced. In our approach, the components used as propagation delay elements are inverters, multiplexers, demultiplexers and their connecting wires. The inverters  $I_{kl}$  ( $k = 0, \dots, (K - 1); l = 0, \dots, (L - 1)$ ) are arranged in form of an inverter matrix  $M_{K \times L}$ . They can be interconnected to form two delay lines that run from the first to the last column of the inverter matrix, using one inverter in each column. These two *Delay Lines* denoted by  $DL_i$  and  $DL_j$ , each with  $L$  inverters, constitute a *Delay Line Pair* denoted by  $DLP_{ij}$ . In order to obtain as many different DLPs as possible, interconnecting *Commutation Circuits* (CCs) are inserted between all columns of the inverter matrix (see Fig. 5). Their internal structure for  $K = 4$  is depicted in Fig. 6a. The same structure can be realized for



**Fig. 5.** IC-EK generator with  $K \times L$  inverter matrix. The delay lines are configured by the commutation circuits  $CC_i$  ( $i = 1, \dots, (L - 1)$ ) and fed back using additional commutation circuit  $CC_0$  to form two ring oscillators. Resulting frequencies are counted by counters  $C1$  and  $C2$ . Response-bit  $\beta_{ij}$  is generated by comparing the counter values.



**Fig. 6.** (a): Internal design of a commutation circuit  $CC_i$  between inverter matrix columns. (b) The two halves of commutation circuit  $CC_0$  that are added before and after the inverter matrix (for  $K = 4$ ).

all other values of  $K$ . The CCs are capable of connecting any pair of inverters of one column to any pair of inverters of the following column allowing only one-to-one connections. Which inverter pair in a column, called *Column Inverter Pair* (CIP), will be selected depends on the control signals of the CCs. In this way  $\Pi = (K^L(K - 1)^L)$  different DLPs can be configured throughout the inverter matrix. As in case of the delay circuit with binary switching components ( $K = 2$ ) in Chapter 3.2, it is essential that the layout of the whole circuit is completely symmetrical. No single delay line may be favored. As can be seen in Fig. 6, the CCs have a completely symmetrical structure, so that they will not cause asymmetry in the layout. The wires that conduct the signals on the delay lines, including those inside the CCs, must be of exactly the same length, regardless

of which combination of delay elements is traversed. No single delay line may be favored by the layout. Of course, in reality this is not possible as the circuit layout is subject to the same random process variations as the delay elements, but those variations will only contribute to the random delay variations of the whole delay lines that are exploited in this design.

To benefit from the accumulation of delay variations in a self-oscillating loop, the outputs of the two configured delay lines are fed back to their inputs to form two *Ring Oscillators* (ROs) as shown in Fig. 5. For oscillations to occur, the number of inverters in each RO must be odd. Either this results in an inverter matrix  $M_{K \times L}$  with odd values of  $L$ , or an additional inverter must be inserted into the feedback lines for matrices with even values of  $L$ . To implement the feedback of the signals, additional circuits are added in front of the first column and after the last column of the inverter matrix (see Fig. 5). They are basically the two halves of one single CC, referred to as  $CC_0$  as depicted in Fig. 6b.

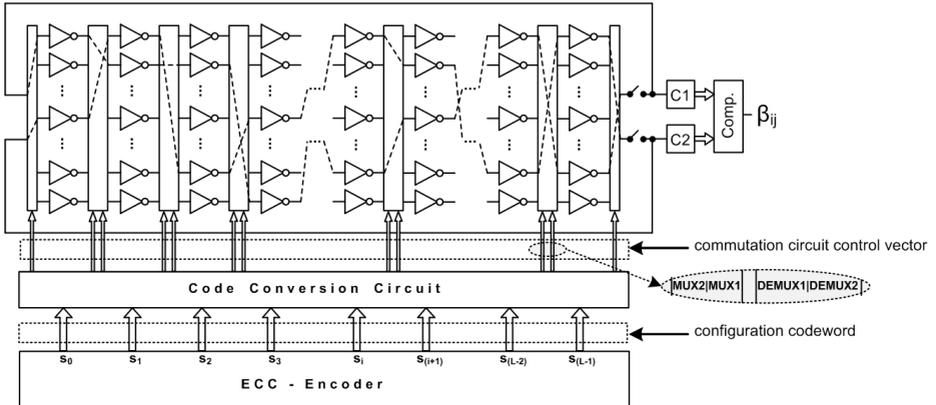
Tests have shown that two ROs oscillating at close range of each other within one IC tend to synchronize their frequencies and oscillate in unison. To prevent this undesirable behavior, one of the ROs could be extended with additional delay elements until its frequency is sufficiently different from that of the other RO to avoid synchronization. However, this modification must be compensated when the difference of the frequencies is measured. Therefore, another solution to this problem will be preferred, where the frequencies of the two ROs will be measured consecutively, interrupting one while the other is oscillating.

In order to ensure that the ROs configured by the CCs are neither interrupted nor fused, certain requirements must be met by the control input bits of the CCs. As shown in Fig. 6, the control input bits will be split into the subsets  $MUX2$ ,  $MUX1$ ,  $DEMUX1$  and  $DEMUX2$ . The whole set of all control input bits of all CCs throughout the inverter matrix will be called the *CC control vector*, as shown in Fig. 7. To configure appropriate ROs, the partial control inputs of each  $CC_l$  before each column (stage)  $l$  of the inverter matrix must satisfy the following equations:

$$\forall_{l=0}^{(L-1)} : (MUX1_l \neq MUX2_l) \wedge (DEMUX1_l \neq DEMUX2_l) \quad (2)$$

$$\forall_{l=1}^L : (MUX1_{(l \bmod L)} = DEMUX1_{(l-1)}) \wedge (MUX2_{(l \bmod L)} = DEMUX2_{(l-1)}) \quad (3)$$

The first equation is necessary, because if the MUX-signals of one stage  $l$  were equal, one delay line would run into a dead end, while the other would be split in two. Furthermore, if the DEMUX-signals of one stage were equal, the two separate paths would be joined into one. Equation (3) ensures, that the CC of each stage picks up the signal from the inverter in the same row to which the preceding CC had passed it on. So the two delay lines will be continuous and not be interrupted. Moreover, equation (3) results in the characteristic of the CCs, where one of the delay lines is always relayed by the lower multiplexer/demultiplexer through all CCs and the other one always by the upper. This also guarantees that the delay lines will not be crossed when they are fed back, which would result in one single RO of  $2L$  length (like a twisted character “8”).



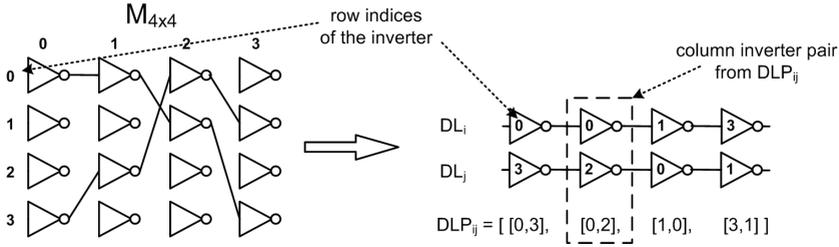
**Fig. 7.** Block scheme of a whole IC-EK generator. An ECC encoder generates the configuration codeword which is translated by a code conversion circuit into the configuration circuit control vector. The oscillations counted consecutively by counter  $C1$  and  $C2$  will be compared to generate a response bit  $\beta_{ij}$ .

With the circuit design described above,  $\Pi = (K^L(K-1)^L)$  different pairs of delay lines  $DLP_{ij}$  can be configured to form the two separate ring oscillators. To evaluate the resulting delays and generate a response bit, their oscillations will be counted by two counters for a fixed amount of time. After that a comparator sets the output bit  $\beta_{ij}$  to either 1 or 0, depending on which oscillator has been faster. So, the value of  $\beta_{ij}$  depends directly on the random variations of the propagation delays of the inverters, gates in the CCs and wires used for the delay lines. In this way, a total of  $\Pi$  bits (one per DLP) can be generated. They form a pool, from which the IC-EK can be built.

As high demands on the statistical independence of IC-EK bits are made in Chapter 2, the issue of which bits to use from this pool is critical. Therefore in the next chapter it will be considered which procedure is suitable to select DLPs for the generation of IC-EK-bits.

## 5 Configuration Control of the IC-Eigenkey Generator

The high degree of statistical dependence of any two response bits generated by the silicon PUFs described in Chapter 3.2 is due to the fact that the delay lines that generated the respective bits shared all their delay elements, only in different combinations. The more delay elements can be used by only one of the two delay line pairs (DLPs), i.e. the more distinctive the difference between them, the less statistical dependent the resulting bits will be. A reasonable assumption is, that the propagation delay variations of inverters combined with the corresponding gates in the CCs are identically independently distributed Gaussian random variables. Under this assumption, two DLPs that consist of completely disjoint sets of inverters from  $M_{K \times L}$  will produce two random, statistically independent



**Fig. 8.** Notation of column inverter pairs and delay line pairs from an inverter matrix  $M_{4 \times 4}$

bits. In order to be able to determine, which DLPs from  $M_{K \times L}$  are best suited to produce a sequence of  $N$  bits as independent as possible, a measure for the difference of two DLPs should be defined.

To this end, a special notation for the description of the DLPs is introduced. As each delay line spans through the whole width of the inverter matrix, using exactly one inverter per column, it is sufficient to denote the index of the row of the used inverter to uniquely identify each delay line. The two delay lines depicted in Fig. 8 would thus be denoted as  $DL_i = [0, 0, 1, 3]$  and  $DL_j = [3, 2, 0, 1]$ .

The two inverters used by a DLP in one column of  $M_{K \times L}$  will be called a *Column Inverter Pair* (CIP) and denoted by the tuple of their row indices. The order in which the indices appear in the tuple is important, as it defines to which delay line in the pair the corresponding inverter belongs. So the CIPs  $[i, j]$  and  $[j, i]$  are not the same. A delay line pair can consequently be uniquely described by the sequence of its CIPs throughout the inverter matrix. In the example of Fig. 8 that would be  $DLP_{ij} = [[0, 3], [0, 2], [1, 0], [3, 1]]$ . Note, that according to (2), the two inverters of a CIP must always be different.

With this definition of delay line pairs, a measure for their difference can be introduced. It will be called the *strong Hamming distance* between delay line pairs and denoted by  $d_{H_s}$ . It reaches its maximum, when in each column of the inverter matrix the CIPs of the two delay line pairs are completely disjoint. Be  $G_s$  the number of columns in which the CIPs of the delay line pairs  $DLP_{ij}$  and  $DLP_{mn}$  share at least one inverter, then their strong Hamming distance will be

$$d_{H_s}(DLP_{ij}, DLP_{mn}) = L - G_s \quad (4)$$

As two DLPs require at least  $K = 4$  rows in the inverter matrix in order not to share an inverter,  $d_{H_s}$  between any DLPs in an inverter matrix  $M_{K \times L}$  with  $K < 4$  will always be zero. To provide a measure for the difference in these cases, too, the *weak Hamming distance*  $d_{H_w}$  between delay line pairs will be defined. In contrast to the strong Hamming distance  $d_{H_s}$ , the weak Hamming distance  $d_{H_w}$  tolerates shared delay elements. As in case of the strong Hamming distance, the maximum value of the weak Hamming distance is also  $L$ , but it will only be decreased if the delay line pairs use completely identical CIPs in a column of the

inverter matrix. Be  $G_w$  the number of columns with such identical CIPs, then the weak Hamming distance between two delay line pairs  $DLP_{ij}$  and  $DLP_{mn}$  will be

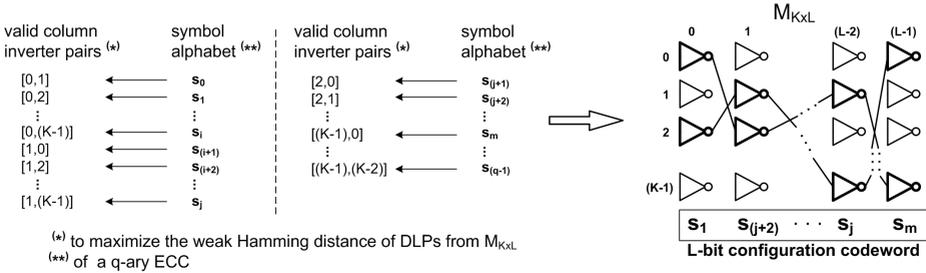
$$d_{H_w}(DLP_{ij}, DLP_{mn}) = L - G_w . \quad (5)$$

As an example, consider the two delay line pairs  $DLP_{ij} = [[0, 3], [0, 2], [3, 0], [3, 1]]$  and  $DLP_{mn} = [[1, 4], [0, 2], [3, 2], [1, 3]]$ . They run through  $L = 4$  stages, from stage  $l = 0$  to stage  $l = 3$ . Their weak Hamming distance  $d_{H_w}(DLP_{ij}, DLP_{mn})$  is  $4 - 1 = 3$ , because only once, in stage  $l = 1$ , do they use the same column inverter pair. Their strong Hamming distance  $d_{H_s}(DLP_{ij}, DLP_{mn})$  is only  $4 - 3 = 1$ , as it is additionally reduced in stage  $l = 2$  where they share one delay element, and in stage  $l = 3$  where the same delay elements are used, only in different order.

As a measure of the difference between two delay line pairs, the weak and strong Hamming distances between them can also be considered a measure of the statistical independence of the two resulting bits  $\beta_{ij}$  and  $\beta_{mn}$ . Only when the strong Hamming distance is maximal ( $d_{H_s}(DLP_{ij}, DLP_{mn}) = L$ ), will the two corresponding random bits be completely independent. The smaller the Hamming distance (strong or weak), the more dependent they are. Given equal values, the strong Hamming distance signifies considerably less statistical dependence between the random bits  $\beta_{ij}$  and  $\beta_{mn}$ .

As the  $N$  bits that form the IC-EK shall be as statistically independent as possible, delay line pairs with a Hamming distance (strong or weak) as large as possible should be chosen for their generation. To realize this selection, *Error Control Codes* (ECCs) with large minimum Hamming distances will be applied. For this, single symbols of a  $q$ -ary symbol alphabet of an ECC can be mapped to the column inverter pairs of the inverter matrix, so that a codeword of length  $L$  of an ECC corresponds to a delay line pair throughout the inverter matrix  $M_{K \times L}$ . The number of codewords of the ECC is equal to the number  $N$  of bits of the IC-EK. An ECC  $(L, N, d_{H_m})_q$  that maximizes the minimal Hamming distance  $d_{H_m}$  between its codewords will be called the *configuration code* of the IC-EK generator. The Hamming distance between two codewords of the configuration code should equal the strong (or weak) Hamming distance between the two DLPs these codewords define.

To appropriately encode the DLPs by the codewords of the configuration code, the *valid* CIPs in  $M_{K \times L}$  will be assigned one-to-one to the code symbols from the  $q$ -ary alphabet of the configuration code. Which of the CIPs are considered valid depends on the level of statistical independence that is to be provided. If it is considered sufficient to maximize the weak Hamming distance between delay line pairs, valid CIPs in an inverter matrix with  $K = 4$  rows would be  $[0, 1], [0, 2], [0, 3], [1, 0], [1, 2], [1, 3], [2, 0], [2, 1], [2, 3], [3, 0], [3, 1]$  and  $[3, 2]$ . An example for this is depicted in Fig. 9. The remaining combinations may not be used because, as stated above, the two elements of a CIP must always be unequal. If instead of the weak Hamming distance the strong one were to be maximized, the number of valid column inverter pairs would in this case be reduced to a set of two, where any two column inverter pairs are disjoint. One set of valid column



**Fig. 9.** Example for mapping a q-ary symbol alphabet of an ECC to the valid column inverter pairs of an  $M_{K \times L}$  inverter matrix, and the delay lines defined by a configuration codeword (valid column inverter pairs to maximize weak Hamming distance between DLPs)

inverter pairs to maximize the strong Hamming distance would be for instance  $([0,1],[2,3])$ . Obviously, the number of possible configuration codewords would be substantially reduced and with it the length  $N$  of the IC-EK. To generate the same number of bits for an IC-EK, a considerably larger inverter matrix would be required. This poses an important design decision, where the tradeoff between IC-EK-security and circuit size must be carefully considered.

A class of ECCs suitable as configuration codes are q-ary linear *Maximum distance separable* block codes, whose minimum distance equals  $p + 1$ , where  $p$  is the number of parity-check symbols [6]. This is the maximal possible value for the minimum distance for given values of  $q, L$  and  $N$ . A subfamily of these codes are the Reed-Solomon codes [30]. They are cyclic, which makes the encoder of these codes simple to realize. However, for this application, they do have certain limitations, e.g. that the number  $q$  of code symbols must be a power of 2 at a codeword length of  $L = q - 1$ . Therefore, modifications of the code (shortening, expurgation or puncturing) might be necessary for certain formats of the inverter matrix  $M_{K \times L}$ . Alternatively, a suitable non linear ECC can be constructed using computer search/optimization algorithms. This task should not be too complex, since the code rate is small and the code length not too high.

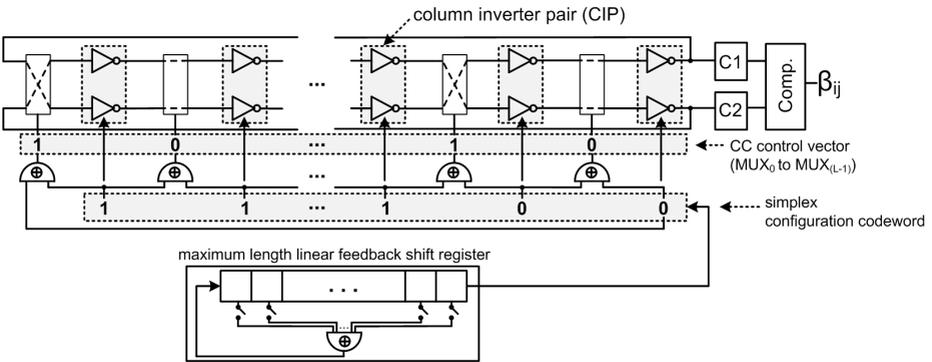
To implement the chosen configuration code with the IC-EK generator, an encoder circuit is needed to generate its codewords. The codewords of the configuration code then have to be converted by a code conversion circuit into the CC control vector needed by the commutation circuits to actually select in each matrix column the CIP that is encoded by the corresponding symbol of the configuration code (see Fig. 7). Note, that in this new application of error control codes neither detection nor correction of erroneous codesymbols are carried out. Instead, the redundancy of deterministic ECCs is used only to select delay line pairs that produce bits as statistically independent as possible. Therefore, no decoder for the ECC is needed, which is a considerable advantage, as an ECC decoder is usually much more complex than an ECC encoder.

A block scheme of a whole IC-EK generator including the ECC encoder and the code conversion circuit is shown in Fig. 7.

## 6 Simulation Results and Tests

An error control code that is especially suited as a configuration code for an IC-EK generator is the *simplex code* [6]. It is a linear binary block code  $(L, L + 1, (L + 1)/2)_{q=2}$  that has a very small code rate  $R = (\log_2(L + 1))/L$ . The number  $N = L + 1 = 2^n$ ;  $(n = 1, 2, \dots)$  of codewords in a simplex code can only be a power of two. This family of error control codes provides the maximally possible minimum Hamming distance  $d_{H_m}$ . All the codewords of a simplex code with codeword length  $L$  have the same Hamming distance  $d_{H_m} = (L + 1)/2$  from each other. This uniform distribution of codewords makes the simplex code ideal as a configuration code for the IC-EK generator.

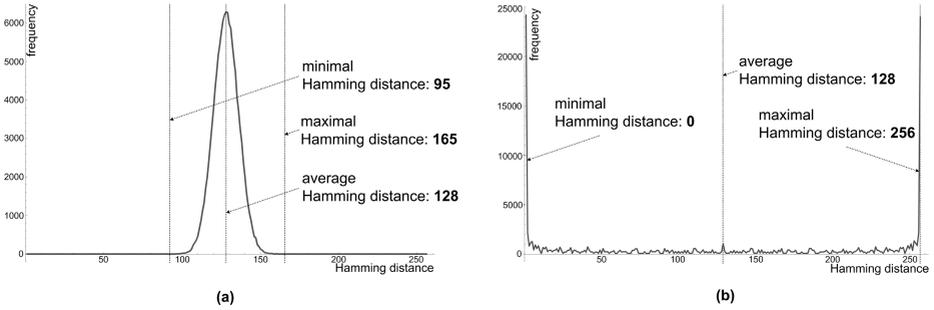
The symbol alphabet of simplex codes is binary,  $q = 2$ . This reduces the number of column inverter pairs (CIPs) that can be encoded, and thus the capability to decorrelate the bits of a resulting IC-EK. Although this could be considered a disadvantage for the application as a configuration code, the simplex code family has the evident advantage that it can be generated very simple by a maximum length *Linear Feedback Shift Register* (LFSR) of length  $n$  [31]. If its feedback is determined by a primitive polynomial, it will produce a different codeword of the simplex code for each of the  $2^n$  possible initial values [32]. An additional advantage of the binary simplex code consists in the very simple code conversion into the CC control vector. This conversion can be realized by  $L$  parallel XOR-gates, as shown in Fig. 10.



**Fig. 10.** IC-EK generator with a  $2 \times L$  inverter matrix. The configuration codeword is generated by a maximum length LFSR and converted by a conversion circuit of parallel XOR-gates into the CC control vector for the CCs.

### 6.1 Simulation Results

In order to check if the ideal uniform distribution of codewords of the simplex code can compensate for the suspected disadvantage of the binary symbol alphabet, a simulation has been performed.



**Fig. 11.** Distribution of Hamming distances of the 256-bit-IC-EKs of 500 simulated ICs with an inverter matrix  $M_{2 \times 255}$ . (a): Using a simplex configuration code. (b): Using Gray-code for configuration.

To keep the duration of the simulation within a reasonable limit, only  $M = 500$  IC-instances have been simulated, each with an IC-EK generator with an inverter matrix  $M_{2 \times 255}$ . The randomly varying delays have been modelled by a Gaussian random variable. The Configuration of the delay line pairs was done using a  $(256, 255, 128)_2$  simplex code. For every single one of the  $M = 500$  simulated IC-EKs of length  $N = 256$ , its Hamming distance to all other IC-EKs has been determined. The distribution of the resulting  $M(M - 1)/2 = 124750$  Hamming distances is presented in Fig. 11a.

For comparison, the same simulated IC-EK generators were also configured using 256 binary 255-Bit numbers, generated by a Gray-code counter initialized with a random starting value. The distribution of the resulting Hamming distances between those simulated IC-EKs can be seen in Fig. 11b. Obviously, in this latter case the Hamming distances that occur by far most frequently are 0 and 256. Most of the resulting IC-EKs are uniformly either 0 or 1, because from one Gray-code number to the next the change in DLP configuration is only marginal (one out of  $L$  CIPs changed). This is not enough to lead to an acceptable probability for a change of the value of the next IC-EK bit  $\beta_{ij}$ . Tests performed using other counters have shown similar results.

In contrast to the counter-based configuration, the distribution of the IC-EKs generated with a simplex code (see Fig. 11a) results in a bell-shaped curve. It shows a minimal value of 95. The same value is estimated using the Gilbert-Varshamov bound given by (1) in Chapter 2. Therefore, it can be assumed that for much larger IC-production series the minimum Hamming distances between all IC-EKs obtained by a simplex code can also be accurately estimated according to the Gilbert-Varshamov bound (1). For the more realistic example at the end of Chapter 1, where  $M$  was approximately 17 millions, a minimum Hamming distance of approximately 80 can be expected. With this Hamming distance, classical privacy amplification (e.g. hash-functions) for additional randomization of the IC-EKs is not needed.

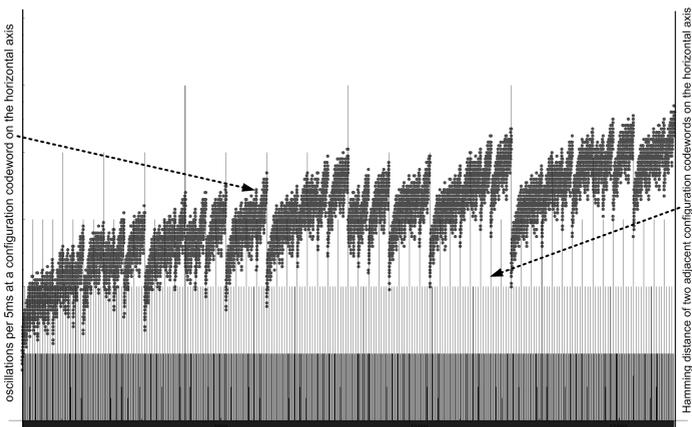
It is interesting to note, that the mean value of the distribution in Fig. 11a is located at  $N/2 = 128$  which is just the minimum Hamming distance of a

simplex code with  $N$  codewords. The codewords of such a simplex code can be regarded as the 256 vertices of a regular simplex polytope in 255-dimensional Euclidian space. The 500 simulated IC-EKs would be randomly located around these vertices, which is ideal to keep an exhaustive search attack after one or more IC-EKs have been compromised as extensive as possible. Therefore, the simplex code family can be considered a very good choice for a configuration code to generate IC-EKs, despite the above presumed disadvantage of a binary symbol alphabet.

## 6.2 Test Results

In order to test a hardware implementation of the IC-EK generator with a simplex code, an  $M_{2 \times 15}$  inverter matrix has been realized with CMOS-gates in separate ICs on a carefully symmetrically designed test board. Two counters implemented on an FPGA-board measured the oscillations of the ring oscillators, while the generation of the codewords of the  $(16, 15, 8)_2$  simplex code and their conversion into CC control vectors were carried out by a microcontroller. The implemented device in this experimental setup schematically corresponds to Fig. 10.

Along the horizontal axis of the diagram in Fig. 12, all possible  $2^{15}$  codewords of the implemented configuration code are uniformly listed. Each dot shows the number of oscillations of one RO during 5ms for a given codeword on the axis. To visualize the impact of the Hamming distance between two codewords on the frequencies of the RO, a bar chart has been added to the diagram. For each codeword on the horizontal axis, the Hamming distance to the one at its left is plotted. It is evident, that larger Hamming distances between the codewords actually result in larger variations of the RO frequencies, which further emphasizes



**Fig. 12.** Oscillation frequencies of one RO of the tested IC-EK generator, compared to the Hamming distances of their configuration codewords

the effectiveness of error control codes with a high minimum Hamming distance as configuration code.

Although the performed test using CMOS-gates in separate ICs was only done at a small scale ( $K = 15$ ), it was nevertheless necessary in order to provide a possibility for measurements at various points of the test equipment, which is not possible in a fully integrated circuit. In this way, various side effects that might influence IC-EK generation could be checked. With this experience, the next step in testing would be to implement the IC-EK generator in a single IC with a high integration scale.

## 7 Conclusion

A method for the protection of sensitive security parameters in integrated circuits is proposed. The sensitive security parameters are symmetrically encrypted using a special secret key, called the IC-Eigenkey. This key is generated by the integrated circuit itself and therefore unknown to anybody else. The IC-Eigenkey is never stored in non-volatile memory. Instead, the complete information about it is kept in a hidden analog form within the integrated circuit. Only on demand can it be extracted and converted to its binary form. For this purpose a suitable extraction circuit is described, called the IC-Eigenkey generator. It is designed based on a silicon physical uncloneable function and uses the randomly varying propagation delays of inverters, multiplexers, demultiplexers and wires which are symmetrically arranged in form of a matrix. Single bits are generated by comparing propagation delays of selected components from the matrix. To choose bits for the IC-Eigenkey that are as statistically independent as possible, a measure is introduced for the dependence of two generated bits from each other. Based on this measure, the selection of matrix components for the generation of bits is realized by codewords of an error control code. This is done in a way that a larger Hamming distance between two codewords results in less statistical dependence between the two generated bits. The high effectiveness of error control codes for this application is demonstrated by a simulation of an IC-Eigenkey generator that uses a simplex code and by practical measurements.

**Acknowledgments.** The authors would like to thank Willi Geiselmann and Sebastian Kaluza for helpful discussions.

## References

1. FIPS PUB 140-2, Security Requirements for Cryptographic Modules, National Institute of Standards and Technology (2002), <http://csrc.nist.gov/groups/STM/index.html>
2. Lemke, K.: Embedded Security: Physical Protection against Tampering Attacks. In: Lemke, K., Paar, C., Wolf, M. (eds.) Embedded Security in Cars. Springer, Heidelberg (2006)

3. Joint Interpretation Library CC/ITSEC: Integrated Circuit Hardware Evaluation Methodology - Vulnerability Assessment, Version 1.3 (2000), <http://www.bsi.de/zertifiz/itkrit/itsec.htm>
4. Smith, S.W., Weingart, S.: Building a High-Performance, Programmable Secure Coprocessor, Technical Report, IBM T.J. Watson Research Center, P.O. Box. Yorktown Heights NY 10598, USA (Revision of October 16, 1998), [http://www.research.ibm.com/secure\\_systems\\_department/projects/scop/papers/arch.pdf](http://www.research.ibm.com/secure_systems_department/projects/scop/papers/arch.pdf)
5. Blahut, R.: Principles and Practice of Information Theory. Addison-Wesley, Reading (1987)
6. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes. North-Holland, Amsterdam (1977)
7. Gilbert, E.N.: A Comparison of Signalling Alphabets. Bell System Technical Journal 31, 504–522 (1952)
8. Varshamov, R.: Estimate of Number of Signals in Error Correcting Codes, Tech. Rep. 117, Dokl. Akad. Nauk, SSSR (1957)
9. Beth, Th., Lazic, D.E., Senk, V.: The Generalised Gilbert-Varshamov Distance of a Code Family and its Influence on the Family's Error Exponent. In: Proceedings of the International Symposium on Information Theory & Its Applications 1994, Sydney, Australia, vol. 1, pp. 965–970 (1994)
10. Beth, Th., Kalouti, H., Lazic, D.E.: Which Families of Long Binary Linear Codes Have a Binomial Weight Distribution? In: Giusti, M., Cohen, G., Mora, T. (eds.) AAEECC 1995. LNCS, vol. 948, pp. 120–130. Springer, Heidelberg (1995)
11. Beth, T., Lazic, D.E., Kalouti, H.: On the Relation Between Distance Distributions of Linear Block Codes and the Binomial Distribution. Annales des Telecommunications, special issue on Channel Coding 50(9-10), 762–778 (1995)
12. Gassend, B., Clarke, D., van Dijk, M., Devadas, S.: Silicon Physical Random Functions. In: Proceedings of the 9th ACM Conference on Computer and Communications Security (2002)
13. Gassend, B., Clarke, D., Lim, D., van Dijk, M., Devadas, S.: Identification and Authentication of Integrated Circuits. In: Concurrency and Computation: Practice and Experience. Wiley, Chichester (2003)
14. Lee, J.W., Lim, D., Gassend, B., Suh, G.E., van Dijk, M., Devadas, S.: A Technique to build a Secret Key in Integrated Circuits for Identification and Authentication Applications. In: 2004 Symposium on VLSI circuits, pp. 176–179 (2004)
15. Gassend, B., Clarke, D., van Dijk, M., Devadas, S.: Controlled Physical Random Functions. In: 18th Annual Computer Security Applications Conference (ACSAC 2002), p. 149 (2002)
16. Gassend, B.: Physical Random Functions, Master's Thesis, Massachusetts Institute of Technology (2003)
17. Gassend, B., Clarke, D., van Dijk, M., Devadas, S.: Delay-Based Circuit Authentication and Applications. In: Proceedings of the 2003 ACM symposium on Applied computing, Melbourne, Florida, pp. 294–301 (2003)
18. Lim, D.: Extracting Secret Keys from Integrated Circuits. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 13, 1200–1205 (2005)
19. Lim, D.: Extracting Secret Keys from Integrated Circuits, Master's Thesis, Massachusetts Institute of Technology (2004)
20. Vrijaldenhoven, S.: Acoustical Physical Uncloneable Functions, Master's Thesis, Technische Universiteit Eindhoven (2004)
21. Suh, G.E., O'Donnell, C.W., Devadas, S.: AEGIS: A Single-Chip Secure Processor. IEEE Design&Test of Computers 24(6), 570–580 (2007)

22. Suh, G.E., O'Donnell, C.W., Sachdev, I., Devadas, S.: Design and Implementation of the AEGIS Single-Chip Secure Processor using Physical Random Functions. In: Proceedings of the 32nd International Symposium on Computer Architecture (ISCA 2005), pp. 25–36 (2005)
23. Pappu, R.S., Recht, B., Taylor, J., Gershenfeld, N.: Physical One-Way Functions. *Science* 297, 2026–2030 (2002)
24. Pappu, R.S.: Physical One-Way Functions. *RSA Laboratories Cryptobytes* 6(2) (Summer 2003)
25. Nassif, S.R.: Modeling and Forecasting of Manufacturing Variations. In: 5th International Workshop on Statistical Metrology, pp. 2–10 (2000)
26. Skoric, B., Maubach, S., Kevenaar, T., Tuyls, P.: Information-Theoretic Analysis of Capacitive Physical Unclonable Functions. *Journal of Applied Physics* 100(2) (2006)
27. Lofstrom, K.: System for Providing an Integrated Circuit with a unique Identification, US Patent Publication, Pat.No. 6,161,213 (2000)
28. Kahlmann, J.A.H.M., Akkermans, A.H.M.: Method for Protecting Information Carrier Comprising an Integrated Circuit, US Patent Application Publication, PUB No. US2007/0038871 A1 (2007)
29. Devadas, S., Gassend, B.: Reliable Generation of a Device-Specific Value, US Patent Application Publication, PUB No. US2006/0271793 A1 (2006)
30. Wicker, S., Bhargava, V.: Reed-Solomom Codes and Their Applications. IEEE Press, Los Alamitos (1994)
31. Bossert, M.: Kanalcodierung, Teubner Verlag Stuttgart (1998) ISBN 3519161435
32. Golomb, S.W., Gong, G.: Signal Design for Good Correlation for Wireless Communication, Cryptography and Radar. Cambridge University Press, Cambridge (2005)
33. Shannon, C.E.: A Mathematical Theory of Communication. *Bell System Technical Journal* 27, 379–423, 623–656 (1948)

# On Reconstruction of RC4 Keys from Internal States

Shahram Khazaei<sup>1</sup> and Willi Meier<sup>2</sup>

<sup>1</sup> EPFL, Lausanne, Switzerland

<sup>2</sup> FHNW, Windisch, Switzerland

**Abstract.** In this work key recovery algorithms from the known internal states of RC4 are investigated. In particular, we propose a bit-by-bit approach to recover the key by starting from LSB's of the key bytes and ending with their MSB's.

**Keywords:** Binary Hypothesis Testing, Stream Ciphers, Key Recovery, RC4.

## 1 Introduction

Synchronous stream ciphers are symmetric cryptosystems which are suitable in software applications with high throughput requirements, or in hardware applications with restricted resources (such as limited storage, gate count, or power consumption). RC4 is probably the most popular stream cipher in use. In this work we are going to investigate the key recovery algorithms from the known internal states of RC4. Roos in 1995 [6] noticed that some of the elements of the initial permutations have a bias towards a linear combination of the secret key bytes. A theoretical proof of these biases was given by Paul and Maitra [5] which was later generalized by Biham and Carmeli [2]. In [5, 2] the authors also provide algorithms for key reconstruction from the internal state using the derived biases. However the algorithms from [5] have high complexities, low success probabilities and the one from [2] has low complexity, and still low success probability. In addition, the authors of [2] did not analyze the complexity of their algorithm for having a higher success probability. In fact, the newly found generalized biases have not been exploited to the degree they deserve in the key recovery algorithm of [2]. The main idea of our work is to *fully* exploit the whole distribution of noises expressing these biases. In a hypotheses testing model, we then study how far one can go by using only the distribution of noises. Having carefully analyzed the noise distributions, we then propose a *bit-by-bit* approach to recover the key bits by starting from LSB's of the key bytes and ending with their MSB's. The nice feature of our algorithm is that we are able to estimate its complexity versus success probability, showing possibility of recovering the key with high success probability but reasonable time complexity.

## 2 Description of RC4 and Notations

RC4 is composed of a Key Scheduling Algorithm (KSA) and a Pseudo Random Generation Algorithm. It works with the set of integers  $\mathbb{Z}_N = \{0, \dots, N - 1\}$  and its internal state is a permutation  $S = (S[0], \dots, S[N - 1])$  over  $\mathbb{Z}_N$  ( $N = 256$  in practice). RC4 uses keys  $k = (k[0], \dots, k[l - 1])$  of length  $l$  (typically  $5 \leq l \leq 16$ ) over  $\mathbb{Z}_N$ . The KSA computes the internal state from the key  $k$  to be used by the PRGA in order to produce a keystream sequence of integers over  $\mathbb{Z}_N$ , see Alg. 1 and 2.

**Notations:**  $K = (K[0], \dots, K[N - 1])$  denotes an array of size  $N$  over  $\mathbb{Z}_N$  such that  $K[i] = k[i \bmod l]$  for  $0 \leq i \leq N - 1$ . The value of the array  $S$  right after the KSA is denoted by  $S_N$  and the array  $C = (C[-1], \dots, C[N - 1])$  over  $\mathbb{Z}_N$  is defined as  $C[-1] = 0$  and  $C[i] = S_N[i] - \frac{i(i+1)}{2}$  for  $0 \leq i \leq N - 1$ . For an array  $A$  we use the notation  $A[i, j] = \sum_{t=i}^j A[t]$ .

---

**Algorithm 1.** RC4 KSA
 

---

```

1: for  $i = 0, 1, \dots, N - 1$  do
2:    $S[i] = i$ .
3: end for
4:  $j = 0$ .
5: for  $i = 0, 1, \dots, N - 1$  do
6:    $j = j + S[i] + K[i]$ .
7: end for
  
```

---



---

**Algorithm 2.** RC4 PRGA
 

---

```

1:  $i = 0$ .
2:  $j = 0$ .
3: repeat
4:    $i = i + 1$ .
5:    $j = j + S[i]$ .
6:   Swap  $S[i]$  and  $S[j]$ .
7:   Output  $z = S[S[i] + S[j]]$ .
8: until enough outputs have been produced.
  
```

---

The goal of the attacker is to determine the secret key out of a known segment of keystream. This can be done in two steps: first to determine a state out from the keystream segment, and in a second step to determine the key out of the internal state. Note that the initial state  $S_N$  can be easily computed given any intermediate internal state at any time during the PRGA. In this paper we only deal with the second step, *i.e.* recovering the secret key  $k$  from a given initial state  $S_N$  (or equivalently from array  $C$ ). The interested reader is referred to [4] for the best known attack on the first step, *i.e.* recovering the internal state from the known keystream segment. Another type of attack on RC4 is key recovery attack when the secret key contains a known initialization vector part and the attacker has access to the keystreams of many (chosen) initialization vectors for the same unknown key part, see [8] for a recent attack.

### 3 Previous Results

Roos in 1995 [6] noticed that some of the elements of the initial permutations have a bias towards a linear combination of the secret key bytes. A theoretical proof of these biases was given by Paul and Maitra [5], later generalized by Biham and Carmeli [2]. Thanks to our choice  $C[-1] = 0$ , these results can be given in a unified theorem as follows.

**Theorem 1.** *Assuming that during the KSA the pseudo-random index  $j$  takes its values uniformly at random from  $\mathbb{Z}_N$ , for every  $-1 \leq i_1 < i_2 \leq N - 1$  then the probability that the following equation holds*

$$C[i_2] - C[i_1] = K[i_1 + 1, i_2] \tag{1}$$

is at least  $p_{i_1, i_2}$  where

$$p_{-1, i_2} = \left(1 - \frac{i_2}{N}\right) \cdot \left(1 - \frac{1}{N}\right)^{\frac{i_2(i_2+1)}{2} + N} + \frac{1}{N} \tag{2}$$

for  $0 \leq i_2 \leq N - 1$  [5] and

$$p_{i_1, i_2} = \left(1 - \frac{i_2}{N}\right)^2 \cdot \left(1 - \frac{i_2 - i_1 + 2}{N}\right)^{i_1} \cdot \left(1 - \frac{2}{N}\right)^{N - i_2 - 1} \prod_{r=0}^{i_2 - i_1 - 1} \left(1 - \frac{r + 2}{N}\right) + \frac{1}{N} \tag{3}$$

for  $0 \leq i_1 < i_2 \leq N - 1$  [2].

In [5] the probabilities in Eq. (2) are used to retrieve the secret key of RC4. The algorithm uses equations of (1) for  $i_1 = -1$  (which in this case is simplified as  $C[i_2] = K[0, i_2]$ ) in the following way. For each combination of  $m$  independent equations out of the first  $n$  equations (*i.e.*  $1 \leq i_2 \leq n$ ), the algorithm exhaustively guesses the value of  $l - m$  key bytes, and solves the  $m$  equations to recover the remaining key bytes. The success of the the algorithm depends on the existence of  $m$  correct and linearly independent equations among the first  $n$  equations. The success probabilities and the running time of the the algorithm for different key sizes and some choices for the parameters  $m$  and  $n$  are presented in Table 1 (taken from [2], see also footnote 1 therein).

In [2] the probabilities in Eq. (3) along with Eq. (1) are used in a more sophisticated way which lead to a better key recovery algorithm. The results can be seen in Table 2. Very recently, Akgün, Kavak and Demirci [1] have developed new biases for RC4, combined them with previous results and provided a new key recovery algorithm from the internal state. It performs better than existing ones including ours. In particular, in a theorem similar to Theorem 1, they provide a lower bound for the probability that  $C'[i_2] - C'[i_1] = K[i_1 + 1, i_2]$  for  $0 \leq i_1 < i_2 \leq N - 1$  where  $C'[i] = S_N^{-1}[i] - \frac{i(i+1)}{2}$  for  $0 \leq i \leq N - 1$ . In this paper we have only used the biases suggested by Theorem 1 since our work had been finalized before having been aware of [1]. Yet [1] *does not*

**Table 1.** Success probability and running time of the algorithm from [5] according to [2]

$l$	$n$	$m$	$P_{succ}$	Time
5	16	5	0.250	$2^{18}$
5	24	5	0.385	$2^{21}$
8	16	6	0.273	$2^{34}$
8	20	7	0.158	$2^{29}$
8	40	8	0.092	$2^{33}$
10	16	7	0.166	$2^{43}$
10	24	8	0.162	$2^{40}$
$l$	$n$	$m$	$P_{succ}$	Time
10	48	9	0.107	$2^{43}$
12	24	8	0.241	$2^{58}$
12	24	9	0.116	$2^{50}$
16	24	9	0.185	$2^{60}$
16	32	10	0.160	$2^{63}$
16	32	11	0.086	$2^{64}$
16	40	12	0.050	$2^{64}$

**Table 2.** Success probability and running time (in seconds) of the algorithm from [2] compared to [5]

$l$	$P_{succ}$	Time [2]	Time [5]
5	0.8640	0.02	366
8	0.4058	0.60	2900
10	0.0786	1.46	183
10	0.1290	3.93	2932
12	0.0124	3.04	100
12	0.0212	7.43	1000
16	0.0005	278	500

fully exploit the distribution of the noises, leaving a room to unify the biases from [1] with those from Theorem 1 in a future work.

### 4 A Hypotheses-Testing Approach to the Problem

Each of the equations (1) gives us a noisy value of  $K[i_1 + 1, i_2]$  which is a linear combination of the key values  $k[i]$ . Let assume that  $e_{i_1, i_2} \in \mathbb{Z}_N, -1 \leq i_1 < i_2 \leq N-1$ , denotes the noise of each equation, *i.e.*

$$K[i_1 + 1, i_2] = C[i_2] - C[i_1] + e_{i_1, i_2} . \tag{4}$$

The random variables corresponding to these noises, denoted by  $E_{i_1, i_2}$ , are not independent (see proofs of Theorem 1 in [5,2]). Moreover Theorem 1 does not completely characterize their probability distribution, since it only suggests  $p_{i_1, i_2} = \Pr\{E_{i_1, i_2} = 0\}$  (we ignore the inequality). However, it is reasonable to assume that  $E_{i_1, i_2}$  takes other values with equal probabilities, *i.e.*  $\Pr\{E_{i_1, i_2} = e\} = \frac{1-p_{i_1, i_2}}{N-1}$  for  $e \in \mathbb{Z}_N^*$ .

With this new view on the problem we try to recover the key in a correlation based attack by taking a hypotheses-testing approach. This can be seen as a generalization of the original correlation attack on binary LFSR’s by Siegenthaler [7]. First, we assume an attacker with an unlimited amount of computational power, capable of making an exhaustive search over all  $N^l$  possible keys. Like [7], we make the assumption that for a wrong guess  $\bar{k}$  (or equivalently the corresponding  $\bar{K}$ ) of the key, the values of  $C[i_2] - C[i_1]$  and  $\bar{K}[i_1 + 1, i_2]$  are uncorrelated. Under this assumption, we are facing the following binary hypothesis testing problem. Given  $N(N+1)/2$  samples of  $e_{i_1, i_2} = \bar{K}[i_1 + 1, i_2] - (C[i_2] - C[i_1])$ ,  $-1 \leq i_1 < i_2 \leq N - 1$ , as a realization of the random variables  $E_{i_1, i_2}$ , decide if the guess  $\bar{k}$  is correct. Our ability in distinguishing between a correct key ( $\bar{k} = k$ ) from a wrong key ( $\bar{k} \neq k$ ) depends on the following two distributions:

$$H_0 : \bar{k} = k, \quad \Pr\{E_{i_1, i_2} = e|H_0\} = \begin{cases} p_{i_1, i_2} & e = 0 \\ \frac{1-p_{i_1, i_2}}{N-1} & e \in \mathbb{Z}_N^* \end{cases} \tag{5}$$

$$H_1 : \bar{k} \neq k, \quad \Pr\{E_{i_1, i_2} = e|H_1\} = \frac{1}{N}, \forall e \in \mathbb{Z}_N . \tag{6}$$

The quality of a decision rule (distinguisher) is related to two kinds of error probabilities: *false alarm probability*  $p_{fa} = \Pr\{H_0|H_1\}$  and *non-detection probability*

$p_{nd} = \Pr\{H_1|H_0\}$ . Ideally, we would like to minimize both error probabilities but normally there is a trade-off. The optimum decision rule is given by *Neyman-Pearson lemma* [3]. It can be shown that for the hypothesis testing problem given by Eq. (5) and (6), the optimum decision rule chooses  $H_0$  if  $M(S_N, \bar{k}) > T$  and selects  $H_1$  otherwise, where

$$M(S_N, \bar{k}) = \sum_{-1 \leq i_1 < i_2 \leq N-1} \log \frac{(N-1)p_{i_1, i_2}}{1 - p_{i_1, i_2}} \cdot \delta(e_{i_1, i_2}) \tag{7}$$

and  $\delta : \mathbb{Z}_N \rightarrow \{0, 1\}$  being the Dirac delta function (*i.e.*  $\delta(e) = 1$  iff  $e = 0$ ). Remember that  $e_{i_1, i_2} = \bar{K}[i_1 + 1, i_2] - (C[i_2] - C[i_1])$  only depends on  $S_N$  and  $\bar{k}$ . The parameter  $T$  determines the false alarm and non-detection probabilities. More precisely we have,

$$p_{fa} = \Pr\{M(S_N, \bar{k}) > T | \bar{k} \neq k\} \tag{8}$$

and

$$p_{nd} = \Pr\{M(S_N, \bar{k}) \leq T | \bar{k} = k\}. \tag{9}$$

### 4.1 Complexity Analysis

Since there are  $N(N + 1)/2$  terms in the sum (7), the complexity of exhaustive search algorithm is  $\frac{1}{2}N(N + 1)N^l$ . As it was noticed in [2] the sum of all the key elements, *i.e.*  $s = k[0, l - 1]$  is quite useful for reducing the complexity. In this section we will show how we can reduce complexity to  $\frac{1}{2}l(l + 1)N^l$ , using  $Nl(l + 1)/2$  memory. The idea is based on the following relations

$$K[i_1 + 1, i_2] = (q_2 - q_1) \cdot s + \begin{cases} \sum_{t=r_1}^{r_2} k[t] & \text{if } r_1 \leq r_2 \text{ \& } (r_1, r_2) \neq (0, l - 1) \\ 0 & \text{if } r_1 = r_2 + 1 \\ -\sum_{t=r_2+1}^{r_1-1} k[t] & \text{if } r_2 + 1 \leq r_1 - 1 \\ s & \text{if } (r_1, r_2) = (0, l - 1) \end{cases} \tag{10}$$

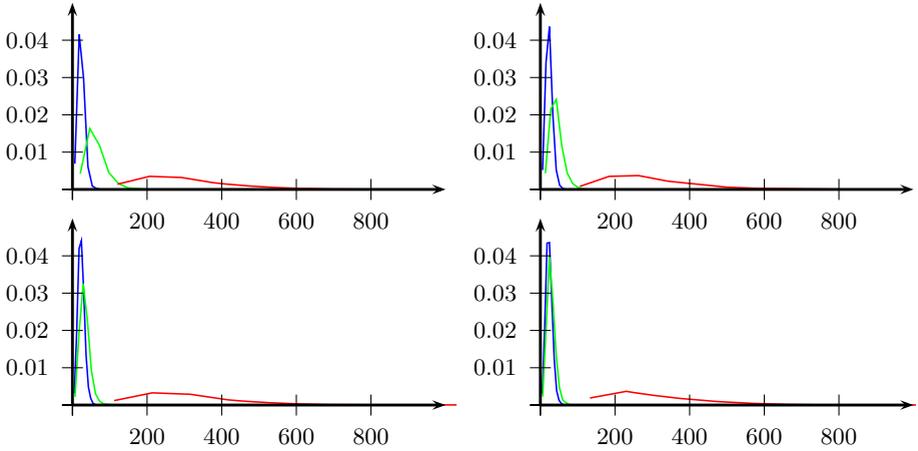
for  $-1 \leq i_1 < i_2 \leq N - 1$  where  $r_1 = (i_1 + 1 \bmod l)$ ,  $q_1 = \lfloor \frac{i_1+1}{l} \rfloor$ ,  $r_2 = (i_2 \bmod l)$  and  $q_2 = \lfloor \frac{i_2}{l} \rfloor$ . Eq. (10) suggests that  $K[i_1 + 1, i_2]$  can be written as  $u(s, i_1, i_2) + \alpha k[r_1, r_2]$  where  $\alpha \in \{-1, 0, 1\}$  and  $u$  being a function of  $s = k[0, l - 1]$ ,  $i_1$  and  $i_2$ . It then follows that Eq. (7) can be written as follows:

$$M(S_N, \bar{k}) = v_{0, l-1}(\bar{s}, S) + \sum_{\substack{0 \leq r_1 \leq r_2 \leq l-1 \\ (r_1, r_2) \neq (0, l-1)}} v_{r_1, r_2}(\bar{k}[r_1, r_2], \bar{s}, S), \tag{11}$$

with  $v_{i_1, i_2}$ ,  $0 \leq r_1 \leq r_2 \leq l - 1$  being some known functions and  $\bar{s} = \bar{k}[0, l - 1]$ . Once  $\bar{s}$  is known, one can precompute and store  $v_{0, l-1}$  and all other  $v_{i_1, i_2}$ 's for all  $N$  possible values of  $\bar{k}[i_1 + 1, i_2]$ . This simply suggests an implementation needing a feasible amount of  $N(l(l + 1)/2 - 1) + 1$  memory and  $N^l l(l + 1)/2$  table look-ups (we ignore the additive time  $N^2(N + 1)/2$  for evaluating  $v_{i_1, i_2}$ 's). Note that  $l(l + 1)/2$  table look-ups are much faster than direct evaluation of Eq. (7) since  $l \ll N$ .

### 4.2 Simulation Results

Our simulation results show that the distributions of  $M(S_N, \bar{k})$ , given by Eq. (7) or (11), under  $H_0$  and  $H_1$  are separated enough to provide a key recovery attack with high success probability. Note that this is a key recovery attack which ONLY takes advantage of the probabilities under our assumptions which are not totally correct (independence of noises and uniform distribution for noise under  $H_1$ ). Moreover, our simulations show that if we consider the hypotheses  $H'_1 : \bar{k} \neq k \ \& \ \bar{s} = s$ , the distributions of  $M(S_N, \bar{k})$  under  $H_0$  and  $H'_1$  get a bit closer but still separated enough, see Fig. 1.



**Fig. 1.** Empirical distribution of  $M(S_N, \bar{k})$  under  $H_0, H_1$  and  $H'_1$  (red, blue and green resp.) for  $l = 5, 8, 12$  and  $16$  (up right, up left, down left and down right resp.)

We introduced  $H'_1$  to slightly compensate the ideal assumption that the noise under  $H_1$  is uniformly distributed. This also helps to estimate a more exact bound for the success probability of a distinguisher which suggests a small set of candidates for the key (*i.e.* the distinguisher having  $p_{fa} \approx N^{-l}$ ). In practice it is not possible to do the simulations for this value of  $p_{fa}$  due to limited number of samples and therefore a practical value should be chosen. Table 3 shows the simulated values for  $p_{nd}$  corresponding to  $p_{fa} \approx 2^{-10}$ , and hence an upper bound estimation for the success probability  $p_{suc}$ .

We expect that the actual success probabilities be very close to our estimations, especially for larger values of  $l$ . The key recovery algorithm of [2] has a much lower success

**Table 3.** An upper bound estimation for the success probability

$l$	5	6	7	8	9	10	12	16
$p_{nd}$ (for $p_{fa} \approx 2^{-10}$ )	0.215	0.125	0.055	0.020	0.018	0.010	0.005	0.000
$p_{suc}$	0.785	0.875	0.945	0.980	0.982	0.990	0.995	1.00

probability though it slightly takes into consideration the dependencies between noises. Also notice that for larger values of  $l$ , the success probability increases which again shows that the algorithm of [2] is far from being optimal.

*Remark 1.* We emphasize that the values of  $p_{\text{suc}} = 1 - p_{\text{nd}}$  (for  $p_{\text{fa}} \approx N^{-l}$ ), give the success probability for an algorithm which makes exhaustive search over the key, only uses the measure  $M(S_N, \bar{k})$  to identify a small subset of candidates for the key, and more importantly does not use the KSA. To achieve a higher success probability one can allow a higher  $p_{\text{fa}}$  resulting in a bigger set of candidates for the key which can later be filtered to find the correct one by applying KSA.

We are interested in algorithms for reconstructing the key which avoid computation of the measure  $M(S_N, \bar{k})$  for all  $N^l$  keys. In the next section we propose an algorithm which starts with an estimate on LSB's of the key bytes and then continues to bits of higher significance.

### 5 A Bit-by-Bit Approach for Key Recovery

The idea is to take into account the probability distribution of  $\Pr\{E_{i_1, i_2} \bmod 2^r\}$  instead of  $\Pr\{E_{i_1, i_2}\}$  and considering these two hypotheses:  $H_0^r : \bar{k} = k \bmod 2^r$  and  $H_1^r : \bar{k} \neq k \bmod 2^r$ . Then one can show that Eq. (5) and (6) become as follows

$$H_0^r : \Pr\{E_{i_1, i_2} = e \bmod 2^r | H_0\} = \begin{cases} p_{i_1, i_2}^r & e = 0 \\ \frac{1 - p_{i_1, i_2}^r}{2^r - 1} & e \in \mathbb{Z}_{2^r}^* \end{cases} \quad (12)$$

$$H_1^r : \Pr\{E_{i_1, i_2} = e \bmod 2^r | H_1\} = \frac{1}{2^r}, \forall e \in \mathbb{Z}_{2^r}. \quad (13)$$

where

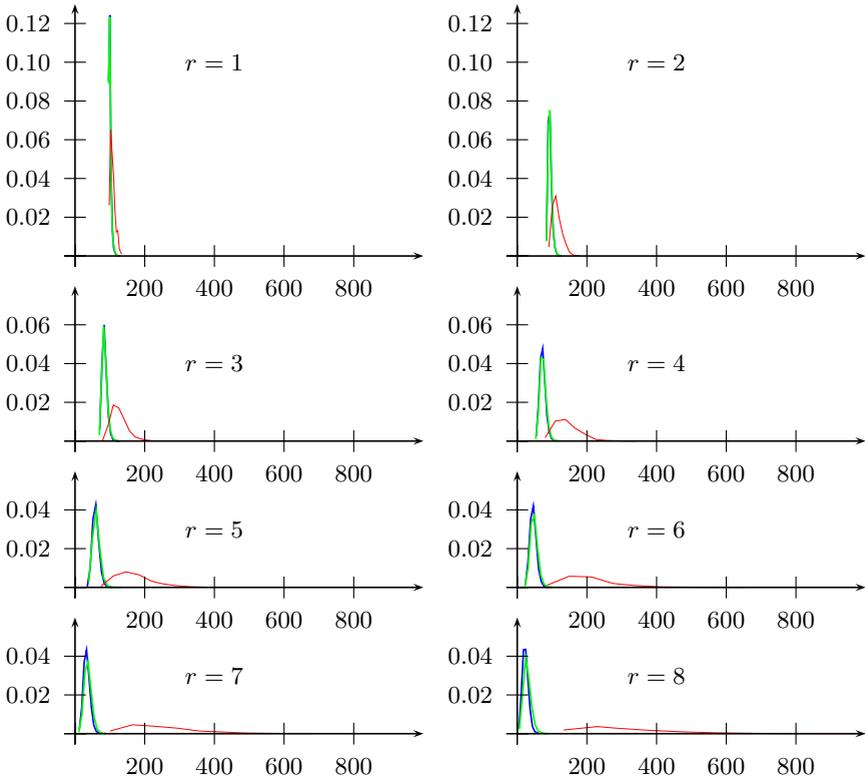
$$p_{i_1, i_2}^r = p_{i_1, i_2} + \frac{N - 2^r}{2^r(N - 1)}(1 - p_{i_1, i_2}) \quad (14)$$

assuming  $N$  is a power of 2. Similarly, the measure which should be computed is as below

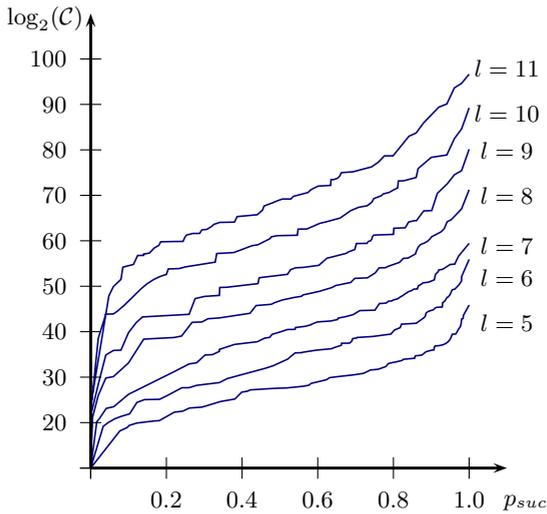
$$M^r(S_N, \bar{k}) = \sum_{-1 \leq i_1 < i_2 \leq N-1} \log \frac{(2^r - 1)p_{i_1, i_2}^r}{1 - p_{i_1, i_2}^r} \cdot \delta(e_{i_1, i_2} \bmod 2^r) \quad (15)$$

having related  $p_{\text{fa}}^r$  and  $p_{\text{nd}}^r$  similar to those in Eq. (8) and (9). Note that  $M^r(S, \bar{k})$  only depends on the first  $r$  LSB's of  $\bar{k}$ .

Fig. 2 shows the empirical distribution of  $M^r(S_N, \bar{k})$  (for  $r = 1, 2, \dots, 8$ ) under three hypotheses  $H_0^r, H_1^r$  and  $H_1'^r$  for  $l = 16$  ( $H_1'^r$  is defined similar to Sect. 4.2). It is clear that the bigger  $r$  is, the more separable the distributions become. Hence a tree-based search can reduce the complexity with a huge factor. The idea is to search over all  $2^l$  possible values of the  $r$ -th LSB of the key elements, assuming that the first  $r - 1$  LSB's of the key are known, and choose only  $N_r$  out of them with the highest correlation measure  $M^r(S, \bar{k})$ , given by Eq. (15). The complexity of this tree-based search algorithm is  $\mathcal{C} = 2^l(1 + \sum_{i=0}^{R-1} \prod_{j=0}^{R-1} N_j)$  where  $R = \lceil \lg_2 N \rceil$ . The KSA must



**Fig. 2.** Empirical distribution of  $M^r(S_N, \bar{k})$  (for  $r = 1, 2, \dots, 8$ ) under  $H_0^r, H_1^r$  and  $H_1^{r^*}$  (red, blue and green resp.) for  $l = 16$



**Fig. 3.** Empirical complexity for  $l = 5, \dots, 11$  versus success probability

be applied to the  $\prod_{i=0}^{R-1} N_r$  key candidates which have reached the final leaves of the tree in order to identify the (possibly) correct one. The complexity of the second step is negligible compared to the first step. The success probability of the attack relies on the parameters  $N_r$ 's. Fig. 3 shows the complexity of the attack versus success probability for optimized parameters of the attack, for different values of  $l$ . Refer to Appendix A to see how these curves have been achieved.

## 6 Conclusion

A recent statistical weakness in the key initialization of RC4 was used to efficiently recover the key from the internal state. We started by *fully* exploiting the whole distribution of noises expressing these newly found biases in RC4 in a hypotheses testing model. Having carefully analyzed the noise distributions, we proposed a tree-based bit-by-bit approach to recover the key bits. It turned out that the complexity of our algorithm can be empirically computed versus its success probability. Further work is still open thank to the more recently developed biases from [1] which we did not exploit.

## References

1. Akgün, M., Kavak, P., Demirci, H.: New Results on the Key Scheduling Algorithm of RC4. *Indocrypt* (to appear, 2008)
2. Biham, E., Carmeli, Y.: Efficient reconstruction of RC4 keys from internal states. In: Nyberg, K. (ed.) *FSE 2008*. LNCS, vol. 5086, pp. 270–288. Springer, Heidelberg (2008)
3. Cover, T., Thomas, J.A.: *Elements of Information Theory*. Wiley series in Telecommunication. Wiley, Chichester (1991)
4. Maximov, A., Khovratovich, D.: New state recovery attack on RC4. In: Wagner, D. (ed.) *CRYPTO 2008*. LNCS, vol. 5157, pp. 297–316. Springer, Heidelberg (2008), <http://eprint.iacr.org/2008/017>
5. Paul, G., Maitra, S.: Permutation after RC4 key scheduling reveals the secret key. In: Adams, C., Miri, A., Wiener, M. (eds.) *SAC 2007*. LNCS, vol. 4876, pp. 360–377. Springer, Heidelberg (2007), <http://eprint.iacr.org/2007/208.pdf>
6. Roos, A.: A Class of Weak Keys in the RC4 Stream Cipher. In: Two posts in *sci.crypt*. (1995), <http://marcel.wanda.ch/Archive/WeakKeys>
7. Siegenthaler, T.: Decrypting a class of stream ciphers using ciphertext only. *IEEE Transactions on Computers* C-34, 81–85 (1985)
8. Vaudenay, S., Vuagnoux, M.: Passive-only key recovery attacks on RC4. In: Adams, C., Miri, A., Wiener, M. (eds.) *SAC 2007*. LNCS, vol. 4876, pp. 344–359. Springer, Heidelberg (2007)

## A Deriving Optimized Parameters for Tree-Based Bit-by-Bit Search Algorithm

In order to derive the complexity diagram versus success probability, Fig. 3, for the optimized parameters  $N_r$ 's for the tree-based bit-by-bit search algorithm in Sect. 5 we

proceed as follows. For every given key size  $l$ , we produce  $\mathcal{N}$  random keys  $k^i = (k^i[0], \dots, k^i[l-1])$ ,  $1 \leq i \leq \mathcal{N}$ . Then for each key  $k^i$ , we compute a vector  $N^i = [N_0^i, \dots, N_7^i]$  where  $N_r^i$ ,  $0 \leq r \leq 7$ , satisfies  $1 \leq N_r^i \leq 2^l$  and denotes the number of choices of the  $r$ -th LSB of  $\bar{k}$  (out of  $2^l$  possible choices) having a correlation measure  $M^r(S_N, \bar{k})$  greater than  $M^r(S_N, k^i)$  provided that  $\bar{k}[j] = k^i[j] \pmod{2^r}$ ,  $0 \leq j \leq l-1$ . As it was mentioned in Sect. 5, for given parameters  $N_r$ 's for the bit-by-bit recovery algorithms, the complexity of the attack is  $\mathcal{C} = 2^l(1 + \sum_{i=0}^{R-1} \prod_{j=0}^{R-1} N_j)$  where  $R = \lceil \lg_2 N \rceil$ . The success probability of the attack can then be estimated as the percentage of the samples for which  $N_r^i \leq N_r$ ,  $\forall 0 \leq r \leq 7$ . However, the parameters  $N_r$ 's may not be optimal and a better choice for them may exist having less time complexity while providing the same success probability. In our simulation we chose  $\mathcal{N} = 1000$  and we tried to find the optimal parameters using a simulated-annealing-like procedure.

## B Improved Recovery of Sum of the Key Elements

In [2] a method has been proposed to recover  $s$ , the sum of the key elements. Our simulations show that using the optimal measure  $v_{0,l-1}(\bar{s}, S)$  slightly improves the results. Table 4 gives the probabilities that the measure  $v_{0,l-1}(\bar{s}, S)$  suggest that  $s$  has the  $i$ -th highest measure ( $i = 1, 2, 3, 4$ ) along with results from [2].

**Table 4.** Probabilities that  $s$  has any of the first four highest measure

$l$	Measure	Highest	Second highest	Third highest	Fourth highest
5	$v_{0,l-1}$	0.888	0.041	0.012	0.016
	[2]	0.8022	0.0618	0.0324	0.0195
8	$v_{0,l-1}$	0.641	0.064	0.042	0.023
	[2]	0.5428	0.1373	0.0572	0.0325
10	$v_{0,l-1}$	0.539	0.091	0.044	0.025
	[2]	0.4179	0.1604	0.0550	0.0332
12	$v_{0,l-1}$	0.441	0.070	0.051	0.041
	[2]	0.3335	0.1618	0.0486	0.0287
16	$v_{0,l-1}$	0.279	0.070	0.039	0.026
	[2]	0.2309	0.1224	0.0371	0.0240

## C Potential Improvements

The tree-based bit-by-bit search algorithm still has some potential for improvements. For example one can imagine a path-ranking on the tree according to their correlation measures and start proceeding on tree from the ones with highest correlation measure at each step. Another idea could be just to ignore some branches in middle if their correlation measure is less than some threshold value. Although these techniques can definitely improve the average time complexity for a given success probability, they are harder to analyze. Another way to improve the bit-by-bit search algorithm is to first

recover the sum of the key elements, and then to use the same method. This way the attack complexity reduces by a factor of about  $2^8$ .

One can also consider the problem as an optimization problem and apply the known methods like genetic algorithm, etc. These methods can easily converge to a key  $\hat{k}$  that maximizes  $M(S_N, k)$ . Our simulations show that the achieved value  $M(S_N, \hat{k})$  is almost always much greater than the correct one  $M(S_N, k)$ . The reason is that there is very small fraction of the keys which have a measure greater than correct key. The fraction is so small that they do not show up in the simulation which provides Fig. 1 and as a result we have separated curves. However, once we use optimization algorithms, it always end up with one of these false keys with highest amount of measure  $M$ . Although it is very unlikely that the resultant key  $\hat{k}$  be the same as correct key  $k$ , but they are quite correlated. For example, usually at least one of the elements of  $\hat{k}$  and  $k$  are the same. It is an open question if we can somehow try to end up with the correct key.

# Author Index

Belovs, Aleksandrs	50	Meier, Willi	179
Di Crescenzo, Giovanni	1	Mora, Teo	126
Grassl, Markus	89	Morozov, Kirill	142
Gropp, Harald	27	Orsini, Emmanuela	126
Huber, Michael	18	Overbeck, Raphael	142
Jozsa, Richard	43	Pernas, J.	128
Khazaei, Shahram	179	Pujol, J.	128
Kobara, Kazukuni	142	Rötteler, Martin	70
Kohnert, Axel	31	Smotrovs, Juris	50
Krovi, Hari	70	Villanueva, M.	128
Kurz, Sascha	31	Wuensche, Micaela	157
Lazich, Dejan E.	157		
Lundqvist, Samuel	105		