Symposium in Honor of Bruno Buchberger's 60th Birthday

LMCS 2002

Logic, Mathematics and Computer Science: Interactions

RISC-Linz, Castle of Hagenberg, Austria
October 20-22, 2002

http://www.risc.uni-linz.ac.at/conferences/LMCS2002

Edited by Koji Nakagawa

Symposium in Honor of Bruno Buchberger's 60th Birthday
Logic, Mathematics and Computer Science: Interactions (LMCS 2002)
October 20-22, 2002
RISC-Linz, Castle of Hagenberg, Austria

RISC-Linz Report Series No. 02-60

ISBN 3-902276-03-7 (electronic version)

Web Page: `http://www.risc.uni-linz.ac.at/conferences/LMCS2002/`

Organizers

General Chairs:
Hoon Hong (USA)
Franz Winkler (Austria)

Program Committee Chair:
Deepak Kapur (USA)

Program Committee:
Franz Baader (Germany)
Alan Bundy (UK)
John Cannon (Australia)
Alain Colmerauer (France)
Nachum Dershowitz (Israel)
Vladimir Gerdt (Russia)
Christoph Hoffmann (USA)
Tetsuo Ida (Japan)
Volker Weispfenning (Germany)

Local Chair:
Tudor Jebelean (Austria)

Local Organization:
Betina Curtis (Austria), Hagenberg Congress GmbH

Publicity:
Wolfgang Windsteiger (Austria)

Proceedings/Web:
Koji Nakagawa (Austria)

Sponsors

Wirtschaftskammer Österreich
(The Austrian Federal Economic Chamber)

BM:BWK Bundesministerium für Bildung, Wissenschaft und Kultur
(Federal Ministry for Education, Science and Culture)

Stadt Linz
(City of Linz)

LHF-Linzer Hochschulfonds
(University Fund of Linz)

Preface

This symposium on Logic, Mathematics, and Computer Science (LMCS), 2002, is being held to honor Bruno Buchberger on his 60th birthday for his many contributions to these areas. I am honored to serve as the chair of the program committee of the symposium. I am grateful for this excellent opportunity to thank Buchberger for influencing my research. Special thanks go to the organizers as well as the researchers, who submitted papers, and the participants for making the symposium a success.

The program was selected from 30 submissions which were received on time; many other submissions which arrived after the deadline could not be considered because the program committee was working under an extremely tight schedule. With the help of our colleagues, the program committee members had the difficult task of choosing a program of 21 papers for presentation on October 20th and 21th, that represented topics from mathematics, computer science and logic. Since this symposium is also a celebration, we wanted to provide ample time for the participants to provide informal comments about the impact of Buchberger's contributions on their research careers. We also wanted the participants to have the opportunity to interact with Buchberger, his former and current students, as well as his former and current colleagues. As a result, many high quality submissions could not be included in the program for which I apologize. I wish this celebration could have lasted for more than 3 days.

I would like to thank all the reviewers (see the list on the next page) and the program committee for their help and cooperation. Special thanks goes to Koji Nakagawa for installing web pages and putting together the proceedings.

Let us make this a memorable event for Bruno!!!

Deepak Kapur
Albuquerque, NM, USA
Oct. 2002

List of Reviewers

Giuseppa Carra' Ferro
Louise Dennis
Volker Gebhard
Ralf Hemmecke
Alex Heneveld
Hoon Hong
Yukiyoshi Kameyama
Alexander Levin
Anton Leykin
Paulette Lieby
Mircea Marin
Bill McCune
Teo Mora
Chin Wei Ngan
Eugenii Pankratiev
Wilhelm Plesken
Allan Steel
Franz Winkler
Daniel Winterstein
Denis Yanovich
Serguey Zemskov

Contents

Abstracts of Invited Talks

Contributed Papers

Abstracts of Invited Talks

Mathematician-friendly proof-assistants

Henk Barendregt

Nijmegen University, Toernooiveld 1, 6525 ED Nijmegen, The Netherlands
henk@cs.kun.nl, http://www.cs.kun.nl/~henk/

Abstract

Computer Mathematics is a generalization of Computer Algebra, in which the system does not only deal with equations, but with arbitrary mathematical objects. These may include non-computable objects. The way this is done is by representing formal proofs. Although a statement A may not be decidable, the statement $\Gamma \vdash p : A$ (p is a proof of A within situation Γ) is decidable.

All researchers in the field are convinced that some day most mathematicians will use such systems of Computer Mathematics, the estimations as to the time it will take ranges from 10 to 50 years.

In this talk it will be indicated what is at least necessary to become mathematician-friendly.

The Role of Logic and Algebra in Software Engineering

Manfred Broy

Institut für Informatik, Technische Universität München, D-80290 München Germany, broy@in.tum.de, http://wwwbroy.informatik.tu-muenchen.de

Abstract

Software systems are among the most complex artefacts humans have built. Software is large, expensive, and difficult to develop, to understand, and to maintain. The requirements and challenges in the development of software are steadily growing. Building software systems is a costly, laborious process influenced by many factors including technology, methodology, experience, management, economy, and application domain know how. Today building software systems is mainly a craft, which due to their size is a major challenge for management and organisation similar to building pyramids in ancient Egypt.

Nevertheless in its essence software represents a technical artefact that is a mathematical object that can be studied by means of logic and algebra. In contrast to classical mathematics where small and homogeneous but rather intricate formulas are considered software is huge, heterogeneous, and incorporates many different aspects. Therefore particular logical and algebraic theories are needed to deal with the various aspects of software. They have to support in particular structuring and separation of concerns. We demonstrate the usage and value of logic and algebra for modelling various aspects of software systems and show a simple, logical clean framework that is powerful enough to describe complex software systems.

We concentrate on modelling issues in software development since software construction is essentially a modelling task. The most important decisions in software development are decisions that deal with modelling. The better, the more adequate and more powerful the available modelling paradigms are, the easier the program development task is and the better its results are. However, a large complex software system can hardly be described and understood by providing one huge model. Instead a number of partial models are used that describe certain aspects of software systems in so-called views and that are in certain mutual relationships such as in levels of abstraction.

We describe the role of models and views in program development and show

how closely the issue of modelling is related to the logical and mathematical issues. Moreover, we give a comprehensive family of models of aspects of software systems and show how to relate and integrate them.

BOOKS OR BYTES?

Dana S. Scott

Carnegie Mellon University
Pittsburgh, Pennslyvania, USA
dana.scott@cs.cmu.edu
http://www.cs.cmu.edu/~scott

Abstract

The hopes of the last decade for electronic publishing have not been realized: some major publishers have recently reduced their commitments. The hopes for distance learning have not been realized: some major institutions have not been financially successful. And more generally the hopes for ease of use of computers in education and science have not been fully realized: we have too many competing formats, too many difficulties with incompatible operating systems, too clumsy interfaces to machines, and too little organization of information. But do we want electrons to replace books and paper and pencil? If we do want to make a suitable evolution to working in the electronic medium, how can we in the university community formulate rational demands and plans to influence the future?

New Directions in the Foundations of Mathematics

Stephen Wolfram

Wolfram Research, Inc.
100 Trade Center Drive
Champaign, IL 61820-7237, USA
http://www.stephenwolfram.com/

Abstract

I will talk about some of the implications of the ideas and discoveries in my book *A New Kind of Science* for the foundations of mathematics. I will address several questions. How general is mathematics as it has been practiced? Is there something special about the axiom systems that have been investigated historically in mathematics? How does one determine what will constitute an interesting theorem or interesting axiom system in mathematics? What happens if one does empirical metamathematics, looking at the structure of mathematics as it has been practiced? Why has Gödel's Theorem not been more important in practical mathematics? What idealizations of mathematics still capture its essence? Is there an intermediate layer of symbolic computation above general transformation rules, and below specific properties of structures like polynomials, that captures mathematics as it has been practiced? To what extent is mathematics as it has been practiced a reflection of human cognitive abilities?

Towards a SymbolicComputational Philosophy (and Methodology!) for Mathematics

Doron Zeilberger

Department of Mathematics, Hill Center-Busch Campus,
Rutgers University, USA,
zeilberg@math.rutgers.edu, http://www.math.rutgers.edu/~zeilberg

Abstract

Symbolic Computation changed my life (for the better, I hope), and Bruno Buchberger changed Symbolic Computation (for the better, I am sure). Ergo, Bruno Buchberger changed my life (most probably for the better). But these two changes are mere iconic SYMBOLS for how Symbolic Computation will change the lives of ALL mathematicians (by changing the way they DO and THINK about their trade (mostly for the better, I hope)), and the way that future mathematical giants will change Symbolic Computation.

Logic, Mathematics, Computer Science: The Accumulated Thinking Technology of Mankind

Bruno Buchberger

Research Institute for Symbolic Computation
University of Linz, A4232 Schloss Hagenberg, Austria
buchberger@risc.uni-linz.ac.at,
http://www.risc.uni-linz.ac.at/people/buchberg

Abstract

In the past century, logic, mathematics, and computer science have seen dramatic breakthroughs, a profound deepening of insight, and an enormous expansion of knowledge.

However, as a matter of fact, in the practice of the research communities, logic, mathematics, and computer science have very little interaction:

- mathematical logic, for the practice of most mathematicians, does not have any noticeable impact,
- many computer scientist are proud that, basically, they can well get along without any significant mathematics, and
- most of the mathematicians are happy and proud that they do not bother about the computer except for reading e-mails, using LaTeX, and searching for literature over the web.

This continues to amaze me and, frankly, I deeply deplore this situation. In fact, I have the feeling that the gap between the three communities widens instead of becoming smaller. I do not see any theoretical reason for this. Rather I think that, mainly, psychologic and sociologic reasons are responsible for the current separation between the three fields:

- All three fields are difficult and it takes a lot of time and effort to become proficient in any of the three fields. Feeling at home in two of them or all three, at first sight, seems to be next to impossible.
- Logic is the meta-theory of mathematics, i.e. logicians are thinking *about* mathematics, namely mainly about the foundational problems of mathematics. For most mathematicians, logic is not the language *in which* mathematics is carried out.
- Mathematicians, normally, do not see any pressing need to improve the logical / formal quality of their work because the addressee of a mathematician,

normally, is again a mathematician who is supposed to be smart enough to overcome logical deficiencies in the presentation of his colleague. (In contrast, the addressee of software engineers is a dull machine that cannot fill in details or correct mistakes. Thus, the necessity of a good meta-theory, namely software technology, has soon become a practical necessity in software development.)

- Mathematicians, notably "pure" mathematicians, notoriously look down on computer science because they believe that algorithms are nothing else than a busy repetition of trivial mathematics and computer scientists or computer mathematicians are people who are not talented enough for "genuine" mathematics. This is of course based on a misunderstanding: Significant progress in algorithmic mathematics and computer science can only be obtained by going deeper into "pure" mathematics and inventing richer theorems with more difficult proofs.
- Conversely, many computer scientists and software engineers believe they do not need mathematics because never in their lives they encounter integrals, complex functions or differential equations. Mathematics, however, is independent of concrete contents, it is the universal arts of reasoning and a sophisticated expression of reasoning in language. In this perspective, computer science, in particular software science, is just a systematic and very explicit expansion of the innermost goals and techniques of mathematics. Thus, it is shortsighted and just a lack of understanding to believe that computer science does not need mathematics.

Starting from this scenario, in the talk, I will sketch a picture of logic, mathematics, and computer science as the one joint and coherent thinking technology of mankind that

- has accumulated over the centuries,
- is being refined, deepened, expanded in a continuous process of self-application,
- and will be the steady and ever expanding basis for the future of the science and technology based global society.

I will draw various conclusions

- on the interplay between research directions in logic, mathematics, and computer science,
- on the essence of algorithmization and self-trivialization as the implicit goal of mathematics,
- on the role and importance of future mathematical knowledge management on a global scale,
- on future integral curricula for mathematics and computer science,
- on the role of mathematics and computer science in society, and
- on the way mathematicians see themselves and the impact of this self-image on their role in society.

Contributed Papers

High Performance Implementations for the Gröbner Bases Algorithm and the Characteristic Sets Method

Iyad A. Ajwa[1] and Paul S. Wang[2]

[1]*Department of Mathematics and Computer Science, Ashland University, Ashland, OH 44805, USA*

[2]*Department of Mathematics and Computer Science, Kent State University, Kent, OH 44242, USA*

Abstract

This paper presents high performance parallel implementations for the Gröbner Bases algorithm and the Characteristic Sets method. Sources of parallelism in the two algorithms have been investigated. The parallel implementations have been conducted on a network of workstations and their performance has been evaluated. Empirical observations have demonstrated significant gains by employing parallelism in these algorithms and suggest that large speedups can be obtained when reasonable parallelism is exploited.

KEYWORDS: Gröbner Bases, Characteristic Sets

1. Introduction

Over the years, new concepts and results have developed in the area of computer algebra and computer algebraists have made significant contributions to the fields of mathematics and computer science. Among these contributions, two outstanding examples are the theory and algorithms for Gröbner Bases and Characteristic Sets.

The concept of Gröbner Bases (GB) was introduced by Bruno Buchberger in 1965. Buchberger's algorithm for computing GB is a powerful tool for solving many important problems in polynomial ideal theory. The algorithm was developed in the context of Buchberger's work on performing algorithmic computations in residue classes of polynomial rings.

Developed from J. F. Ritt's work on differential algebra, the Characteristic Sets (CS) method was discovered independently by Wen-tsün Wu in 1978. Wu rediscovered the CS method in the context of his work on mechanical geometry

theorem-proving. He developed Ritt's work for the algebraization of geometry and introduced a powerful algebraic algorithm to compute CS.

The concepts of GB and CS are important in terms of theory and have found valuable practical applications in mathematics and computer science. Specific applications include polynomial system solving, automated geometric reasoning, CAD/CAGD, robotics, and computer vision. The two algorithms have been extensively studied, developed, refined, and have been implemented on most computer algebra systems.

1.1. Parallel GB and CS

The study of parallel GB and CS has developed into a new area of computer science. Paralleling the GB algorithm and the CS method is a nontrivial undertaking. Tools for symbolic mathematical manipulation, inter-process communication, and controlling parallel/distributed tasks are needed. This paper focuses on applying parallelism to the GB algorithm and the CS method.

1.1.1. Motivation

Speed and efficiency are crucial factors in our field because real-world problems often involve huge symbolic and algebraic computations. In spite of great efforts over the years, sequential computations of GB and CS are still time and space consuming. Algebraic algorithms like these, which are universal for a broad class of problems, are very compute intensive. High speed is, therefore, essential for using these algorithms in practical cases. Based on analysis and experiments, we believe that the power of the GB algorithm and the CS method could be enhanced by use of parallel processing. The two algorithms contain natural sources of parallelism. Further, they produce a lot of mutually independent subproblems that may be treated in parallel.

1.1.2. Related Work

In 1985, Buchberger (3) was the first to propose a parallel algorithm to compute GB. Since then, there have been several attempts to implement the GB algorithm in parallel. We refer the reader to (1) for a comprehensive list of these attempts. Several of these attempts were successful but reported limited speedup. Perhaps the most successful parallel implementations of the GB algorithm are due to Faugère (5) and Amrhein-Gloor-Küchlin (2).

In 1991, Dongming Wang (7) presented a parallelized version of the CS method. He implemented the parallel algorithm using a Maple research system and reported a speedup of 5 using 12 processors over the sequential version of the algorithm. These are the only experiments reported in the literature regarding parallel implementations of the CS method.

1.1.3. Ongoing Research

We present new parallel implementations for the GB and CS algorithms that use message passing in a master/slave paradigm and have the following features:

- Parallelism exploited is natural and quite obvious.
- The implementations demonstrate that large speedups are possible when reasonable parallelism is applied.
- The parallel implementations can be installed on standard networks and rely only on publicly available software making them widely available and independent of commercial computer algebra systems.
- The parallel implementations are based on the most efficient sequential implementations available: GRÖBNER (8) and CSETLIB (9) which are written in the **C** language and are based on the SACLIB (4) system.

The sequential version of the GB algorithm is illustrated in Figure 1. Polynomial reduction is the corner stone of the GB algorithm. It is the most computationally intensive part of the algorithm. Analysis and experiments have shown that most of the time is spent in reducing the *S-polynomial*s. During this time, neither the basis nor the set of pairs change. This fact allows several reductions to proceed simultaneously and independently. We follow the common approach to parallelizing the GB algorithm which has been the parallelization of the main loop shown in Figure 1. In our approach, we compute and reduce the *S-polynomial*s in parallel using a message-passing model within a master/slave paradigm. This scheme is illustrated in Figure 3.

For the CS method, our parallelization strategy focuses on parallelizing the inner loop of the main loop of the algorithm illustrated in Figure 2. The basic operation in the CS method is the pseudo-division of polynomials. Analysis and experiments have shown that many costly pseudo-divisions are needed. The formation of the set RS is thus the most time consuming task, especially when the intermediate polynomials become large. The inner loop of the main loop performs the computations to form RS and is a prime candidate for parallelization. In each iteration of the inner loop, a polynomial from the set $PS = QS - CS$ is picked and a *pseudo-reduction* with respect to CS is carried out. If the *pseudo-remainder* is non-zero, it is adjoined to the set RS. The inner loop terminates when all polynomials from PS are pseudo-reduced. During the pseudo-reduction process, QS and CS remain unchanged. The strategy is to divide PS into several disjoint subsets of polynomials. All subsets are then pseudo-reduced in parallel. This scheme is illustrated in Figure 4.

Our approach to paralleling the GB and CS algorithms is straightforward. We took the most efficient sequential implementations reported in the literature and applied parallel/distributed computation to them. This application resulted in two programs for each algorithm: a master program and a slave program. The master program performs a special task: coordinating the work of the slave programs running on multiple nodes of the network. The slave programs all perform

the same task: the actual computations and reductions. Each slave program communicates only with the master program in a star topology. Slaves and master programs communicate data via PVM and SaclibPvm as explained in Section 2. Assuming the availability of $n+1$ processors at our disposal, we run the master program on one processor and we run one slave program on each of the remaining n processors. We discuss details of implementations in Section 3. But first, we discuss tools built for the parallel implementations.

2. Parallel Implementation Tools

Paralleling the GB algorithm and the CS method is a nontrivial undertaking. Tools for symbolic mathematical manipulation, inter-process communication, and controlling parallel/distributed tasks are needed. For these purposes, we used SACLIB, GRÖBNER, CSETLIB, PVM, SaclibPvm, PvmJobs, and a set of PVM enhancements.

2.1. The SaclibPvm Interface

SaclibPvm is a simple software package interfacing SACLIB to PVM. Although there are different paradigms for performing parallel processing, all have one thing in common: processes need to exchange data. For our work, we need parallel tasks to exchange mathematical data such as large integers and polynomials in one or more variables. PVM provides mechanisms for exchanging basic data types such as integers, float numbers, and strings. A parallel implementation that uses SACLIB, however, needs to exchange SACLIB data. The SaclibPvm is an interface we built that enables any parallel tasks to transfer SACLIB data via PVM. The routines were written in **C** using basic PVM and SACLIB functions and provide mechanisms for packing, unpacking, sending, receiving, and multicasting SACLIB data.

2.2. The PvmJobs Mechanism

PvmJobs (6) is a generic parallel jobs library for PVM. It is a general bag-of-jobs library that works with any user created job structure in a master/slave paradigm. PvmJobs also provides a simple job scheduling mechanism to distribute jobs among the slave processes. It allows the user to substitute PvmJobs's scheduling scheme with any users-defined priority-driven scheduling. Early versions of PvmJobs was written by the authors. Recently, Hong Ong (6) has extended PvmJobs by adding new features and making the library more general and easier to use. The library is written in **C** and has a set of user level functions which provide a flexible API.

3. Parallel Implementations

Parallel implementations have been conducted on a network of workstations. Details of the implementations of the GB algorithm and the CS method are

presented in Section 3.2 and Section 3.3 respectively. But first, we present the sequential versions of the GB and CS algorithms.

3.1. Sequential Algorithms

Input: A set of multivariate polynomials F
Output: A Gröbner Basis, G
Method:
$G := F$
$Pairs := \{\{f_i, f_j\} | f_i, f_j \in G \text{ and } f_i \neq f_j\}$
Repeat
    $\{p, q\}$:= a pair in $Pairs$
    $Pairs := Pairs - \{\{p, q\}\}$
    $S :=$ *S-polynomial(p, q)*
    $h :=$ *NormalForm(S, G)*
    IF $h \neq 0$ THEN
        $Pairs := Pairs \bigcup \{\{g, h\} \forall g \in G\}$
        $G := G \bigcup \{h\}$
Until $Pairs = \emptyset$.

Figure 1: Sequential Gröbner Bases

Input: A set of multivariate polynomials F
Output: A Characteristic Set, CS
Method:
$QS := F$;
$RS := F$;
Repeat
  $CS := BasicSet(QS)$;
  IF CS is contradictory then
      $RS := \phi$
  ELSE
      $RS := \{r \mid r = PseudoRemainder(q, CS), r \neq 0, q \in QS - CS\}$;
      $QS := QS \bigcup RS$;
Until $RS = \phi$

Figure 2: Sequential Characteristic Sets

3.2. Parallel GB

We now describe concrete implementation schemes for parallelizing the GB algorithm based on the strategies discussed in Section 1.1.3.

The master process performs a special task: coordinating the work of the slave processes. The slave processes all perform the same task: the actual computations and reductions. Each slave communicates only with the master in a *star topology*. Slaves and master processes communicate data via PVM and SaclibPvm as explained earlier.

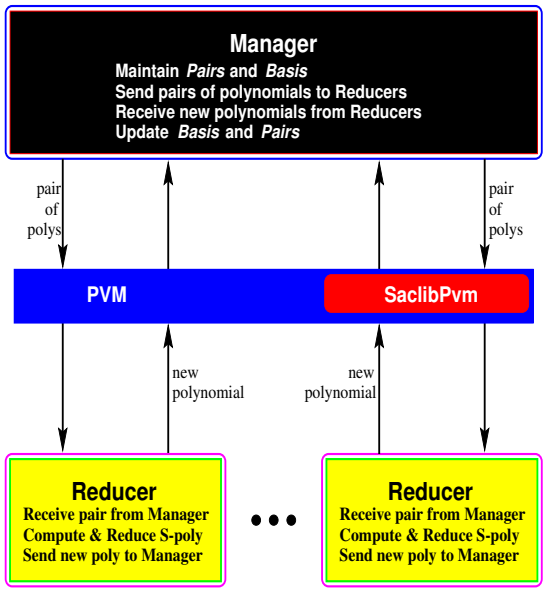


Figure 3: Parallel Gröbner Bases

3.2.1. Parallel GB: Master Process

The master program performs control functions: spawning slave processes and setting up the bag-of-jobs; a loop of sending jobs to slave processes, receiving answers from slave processes, and updating bag-of-jobs, basis, and the set of pairs; and a cleanup process including freeing space reserved for SACLIB and exiting PVM.

The master process, `gb_master.c`, and the slave process, `gb_slave.c`, both begin by including the necessary header files followed by their own variable declarations, initializing SACLIB, and then enrolling in PVM.

The first step taken by the master process is spawning the slave task, by calling PvmJobs library function `job_init` and getting input which consists of a set of polynomials.

The master program then initializes Gröbner basis, forms the set of pairs, and sets up the bag-of-jobs.

The master program is now ready to begin computing Gröbner basis for the given set of polynomials. The program enters a loop of sending pairs of polynomials to slave processes who are up and ready. It receives returning polynomials which will be either zero or non-zero polynomials. In case returning answer is a non-zero polynomial, the master program updates the `Basis`, by adjoining the new polynomial to `Basis`, and the set of pairs, by forming new pairs of polynomials as described above. The bag-of-jobs is also updated by adding new jobs. The loop does not terminate till all jobs farmed out to slave processes have come back and the master program is out of jobs to farm out.

When the loop terminates, the master program stops the timer, displays the computed Gröbner Basis and timing data. It then does cleanup, exits PVM, exits SACLIB freeing reserved space, and terminating any slave processes who are still up and waiting for jobs.

3.2.2. Parallel GB: Slave Process

The slave program begins with enrolling in PVM and requesting its first job. Then the program enters an infinite loop of receiving a pair of polynomials from the master program, computing and reducing the *S-polynomial* of the pair, and sending the normal form of the computed *S-polynomial* back to the master. The slave process breaks out of the loop and terminates itself when a special NULL job is received from the master process.

3.3. Parallel CS

The basic operation in the CS method is the pseudo division of polynomials as discussed in Section 3.1. Analysis and experiments have shown that many costly pseudo divisions are needed. The formation of the set RS is thus the most time consuming task, especially when the intermediate polynomials become large. The inner loop of the main loop performs the computations to form RS and is a prime candidate for parallelization.

In each iteration of the inner loop, a polynomial from the set $QS-CS$ is picked and a *pseudo reduction* with respect to CS is carried out. If the *pseudo remainder* is non-zero, it is adjoined to the set RS. The inner loop terminates when all polynomials from $QS - CS$ are pseudo reduced. During the pseudo reduction process, QS and CS remain unchanged. The strategy is to divide $QS - CS$ into several disjoint subsets of polynomials. All subsets are then pseudo reduced in parallel.

3.3.1. Load Balancing and Scalability

Balancing the loads of slave processes is a major concern in parallel processing. Ideally, all slaves are kept busy doing useful work all the time till the entire computation is finished. Perfection is hard to achieve, but a good load balancing scheme is important for the efficiency of any parallel algorithm. Load imbalance in parallel CS computations could occur due to the following reasons.

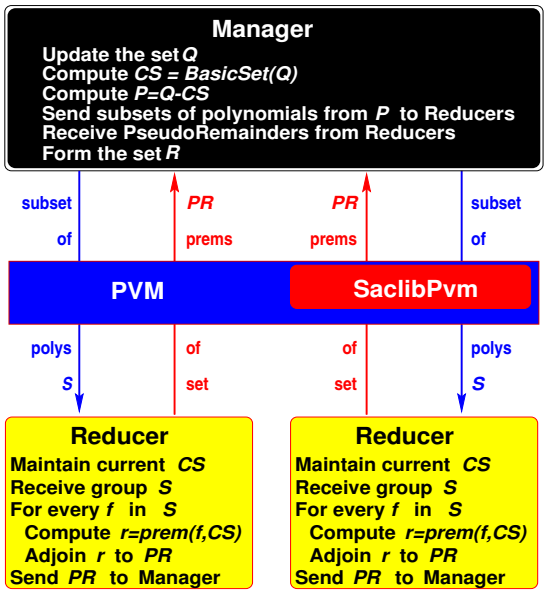


Figure 4: Parallel Characteristic Sets

- If the number of polynomials to be pseudo reduced in parallel is less than the number of slaves, then some slaves will be idle. This situation only arises for small problems where parallelism at this level helps little anyway.
- Different polynomials take different amounts of time to be pseudo reduced. As a result, one or several slaves can be idle while other slaves are busy although they all received the same number of polynomials from the master process.

A flexible load balancing scheme is required in order to achieve good performance. We have considered the following three approaches:

1. A totally dynamic scheduling would send one polynomial from the set $QS - CS$ to each slave when it is ready. This can reduce idle time but may incur too much communication overhead and may cause a bottleneck specially when many slave processes are used.
2. A completely static scheduling scheme would partition the set $QS - CS$ into equally-sized subsets of polynomials and send each subset to a different slave. This approach reduces communication time but may result in unacceptably high idle time since it is hard to predict which polynomials are harder to pseudo-reduce.
3. A hybrid approach would farm out *several (s)* polynomials at once using technique (2) and hold the *remaining (r)* polynomials to dish out using technique (1). The ratio s/r to use depends on the number of slave processes *(n)* and the total number *(t)* of polynomials to be pseudo-reduced. The value

of *(t)* depends on how many intermediate polynomials are generated in the algorithm and is not easily predicted. More experimentation can provide some basis for heuristic values of s/r.

Based on analysis and experiments, we have adopted the third approach for load balancing. Let t be the number of polynomials to be reduced and let n be the number of slave processes. Then the size of each job s to be sent to each slave is t / n using integer division. Hence, the number of jobs to be done is n. Now, the number of remaining polynomials is $m = t$ *MOD* n. Each polynomial of these m polynomials makes-up a job by itself. Hence, the total number of jobs to be performed is $n + m$. It should be noted here that load balancing is applied every time a new set of jobs is formulated. i.e., at the beginning of each iteration of the outer loop.

Before we describe the master and slave programs of the CS method, we present a description of the data structures used in the master and slave processes.

3.3.2. Parallel CS: Master Process

The master program computes and maintains the basic sets, sends copies of the current basic set to slave processes, sends subsets of polynomials to slave processes, receives pseudo-remainders from slave processes, maintains the set QS, and forms the set RS. The job of the master process terminates when forming a new set RS fails.

The master process, `pcs_master.c`, and the slave process, `pcs_slave.c`, both begin by including the necessary header files followed by their own variable assignments, initializing SACLIB, and then enrolling in PVM.

If the input set is not trivial, the master program proceeds to compute the Characteristic Set for the given set of polynomials. The program enters a loop. The first step is to spawn slave processes. The master program then computes the basic set, determines which polynomials need to be pseudo-reduced, initializes the bag-of-jobs according to the load balancing scheme discussed above, and then enters the inner loop for forming the RS set. The inner loop consists of sending subsets of polynomials to the slave processes who are up and ready. It receives returning polynomials which will be either zero or non-zero polynomials. In case a returned answer is non-zero, the master program adjoins those polynomials to RS. The loop does not terminate till all jobs farmed out to slave processes have come back and the master program is out of jobs to farm out. That is when the inner loop exits and a new iteration of the outer loop begins. When the loop terminates, the master program stops the timer, displays the computed Characteristic Set together with timing data. It then does cleanup,exits PVM, exits SACLIB freeing reserved space, and terminating any slave processes who are still up and waiting for jobs.

3.3.3. Parallel CS: Slave Process

Each slave maintains a copy of the current basic set. A slave computes the pseudo-remainders of an assigned set of polynomials with respect to the current basic set (the pseudo-reduction). Once done, the slave sends the computed set of pseudo-remainders back to the master and may receive a new set of polynomials to process. A slave terminates when the master program gives it no more jobs.

4. Conclusion

In this paper, new parallel implementations for the GB algorithm and the CS method have been presented. The performance of the programs varies depending on the size of the problem and lead to the conclusion that the implementations presented in this paper exploit reasonable parallelism and have achieved good speedups.

References

1. I. Ajwa, "Parallel Algorithms and Implementations for the Gröbner Bases Algorithm and the Characteristic Sets Method," Ph.D. Dissertation, Kent State University, Kent, Ohio, December 1998.

2. B. Amrhein, O. Gloor, and W. Küchlin, "A Case Study of Multi-Threaded Gröbner Basis Computation," Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation (ISSAC '96), Y. N. Lakshman (editor), pp. 95-102, Zurich, Switzerland, July 24-26, 1996.

3. B. Buchberger, "The Parallel L-Machine for Symbolic Computation," EUROCAL '85, European Conference on Computer Algebra, B. F. Caviness (editor), Lecture Notes in Computer Science 204, Springer-Verlag, pp. 541-542, Linz, Austria, April 1-3, 1985.

4. B. Buchberger *et al*, *SACLIB 1.1 User's Guide*, RISC-Report No. 93-19, Linz, Austria (1993).

5. J. Faugère, "Parallelization of Gröbner Basis," Proceedings of the First International Symposium on Parallel Symbolic Computation (PASCO '94), H. Hong (editor), Lecture Notes Series in Computing, vol. 5, World Scientific, Linz, Austria, September 1994.

6. H. Ong, I. Ajwa, and P. Wang, "PvmJobs: A generic parallel jobs library for PVM," Proceedings of the 1997 IEEE National Aerospace and Electronics Conference (NAECON'97), (editors), pp. - , Dayton, Ohio, USA, July 14-18, 1997.

7. D. Wang, "On the Parallelization of the Characteristic-Set-Based Algorithms," Proceedings of the 1st International Conference ACPC, 1991.

8. W. Windsteiger and B. Buchberger, "GRÖBNER: A Library for Computing Gröbner Bases Based on SACLIB," RISC-Linz Technical Report No. 93-72, Johannes Kepler University, Linz, Austria, 1993.

9. L. H. Zhi, "Polynomial Factorization over Algebraic Field and Its Applications," Ph.D. Thesis, Institute of Systems Science, Academia Sinica, China (1996).

Solving For Functions

Michael Beeson
San Jose State University
Math & Computer Science
San Jose, CA 95192

Abstract

Buchberger has emphasized that automated deduction involves computation, proving, and solving. When the object to be "solved for" is a function, second-order unification can be a very powerful and general solution tool. An algorithm for second-order unification was given in (8). This algorithm is now being implemented in the source code of Otter; this algorithm differs from the earlier algorithm given by Pietrzykolski (22). Several examples are given to illustrate the wide range of potential applications of this algorithm and its implementation in a powerful clausal theorem prover, including the ability to manipulate quantifiers at the clausal level, so that definitions involving quantifiers can be conveniently used in proofs. Since predicates are considered as Boolean-valued functions, solving for functions includes solving for predicates, and second-order unification can help first-order provers with proofs by induction. Other examples show proofs in which the function to be solved for is a one-to-one correspondence (in set theory) or a group isomorphism. There is also a detailed comparison of the new second-order unification algorithm with the older one.*

KEYWORDS: automated deduction, computer proofs, unification, second-order, Otter

Introduction

In previous work (4; 7), the author has focused attention on the relation between computation and logic in automated deduction. In the development of *Theorema*, Buchberger (13) was guided by his vision that mathematics involves *three* essential components: logic (proving), computation, and *solving*. Solving

*Research supported by NSF grant number CCR-0204362.

means finding an x with certain desired properties. Often the "solving" step of a proof is the "key" step, the one that seems to demand "creativity". Buchberger emphasizes the importance of adding (many) special-purpose "solvers" to a computerized mathematical system. In this paper, by contrast, we wish to focus on *general* solution techniques, as opposed to special algorithms that apply to a specific mathematical theory.

Solving refers to techniques for instantiating a variable. The traditional method of instantiating variables used in automated deduction is unification. It is known that, with respect to first-order logic, this method is "universal", in the sense that there are completeness theorems for various systems of inference in which unification is used to instantiate the variables. For example, binary resolution with unification is complete, and backwards application of the Gentzen sequent-calculus rules, guided by unification, is complete. But when we go to mathematics, unification does not seem to do the job. For example, if we want an x such that $x^2 + ax + b = 0$, we need the quadratic theorem, not unification. Therefore, as Buchberger has emphasized, special-purpose solvers are required to deal with the different specialized branches of mathematics.

Our focus in this paper is on situations where we need to "solve for" a function, rather than an element or object. We will exhibit a number of examples of proofs of this kind, to illustrate the claim that "solving for a function" is a theme that permeates different branches of mathematics. Just as this is a general theme in mathematics, there is a general tool in logic to help with this kind of proof. There is a well-known "unification algorithm" which can be thought of as solving equations between terms denoting objects. There is also λ-calculus, which lets us define terms for denoting functions. There is a way (in fact, more than one way) to generalize the unification algorithm so that it can be thought of as solving equations between terms denoting functions, rather than objects. This is known as *second-order unification*.

Our motivation in this paper is to show that, even though special-purpose solvers are useful for specialized mathematics, we can get some surprising (and in some cases unexpected) results from general methods based on second-order unification.

The problems addressed here are: How are the existing notions of second-order unification related? Can second-order unification be fruitfully used in a first-order theorem-prover? Which is the best notion of unification to use in automated deduction? Is it feasible to add second-order unification to an existing prover (Otter), which already has a large group of users? How can we expect this capability to be useful in automated deduction?

These questions have not been answered until now. One reason for this is that the majority of work in automated deduction has been done (so far) by first-order theorem provers, but second-order unification has so far been (incorrectly) viewed as incompatible with first-order provers. Another reason is that serious automated deduction has so far been done in theories with a single short list of axioms, referring to only one kind of mathematical objects, rather than in more complex mathematical environments, where second-order unification might

prove helpful. A third reason is that second-order unification is considered inefficient (it produces infinitely many unifiers, it necessarily produces redundant unifiers, it involves an exponential search, etc.)

The main points of the paper are

(i) Second-order unification can be thought of as "solving for a function", that is, finding a term that defines a function with desired properties to complete a proof.

(ii) Some mathematical problems that may not appear to have the form of "solving for a function" can be recast in that form, so that second-order unification can be used on them. Others are naturally of that form. A very general "solver" can be useful, because solving for functions occurs generally in mathematics.

(iii) Second-order unification can be implemented and used in Otter, a first-order theorem-prover already widely used (19). There is nothing strange or difficult about using λ-calculus and "second-order" unification in a first-order prover. Sample proofs produced by the new implementation in Otter are exhibited.

(iv) Technical difficulties concerning efficiency, non-termination, redundancy, etc., are minimized if we use the notion of unification introduced in (8) rather than the one in (22; 18). These notions are compared here.

(v) Quantification can be defined in terms of λ-calculus, and second-order unification makes it possible to use quantifiers in Otter proofs at the clausal level. This is very important as it will enable Otter to work with definitions involving quantifiers, such as the relation "u divides v" on the natural numbers.

(vi) Set theory too is naturally treated as a branch of λ-calculus. This is not an original idea but goes back to Church.

We shall give a motivating example. In undergraduate courses in algebra, what is taught as "group theory" usually involves the study of groups and subgroups. See e.g. the first 45 pages of (15) for the mathematics in question. The notation a^n is introduced, where n is a natural number, and one of the early theorems is Lagrange's theorem, according to which the order of a subgroup H of a finite group G divides the order of G. This theorem involves groups, subgroups, and natural numbers. Its proof begins by showing that each coset Ha is in one-to-one correspondence with H. That is, there exists a function f mapping H one-to-one onto Ha. That function, of course, is $\lambda x.xa$. The proof therefore involves a small amount of set theory to deal with cosets and one-to-one correspondences, as well as enough number theory to deal with "divides". We will show in the last section of this paper that second-order unification can instantiate the variable f properly to do the key step of this proof automatically.

Buchberger's aim in developing *Theorema* has been to develop an interactive environment in which humans can develop computer-checked proofs in a mathematical context like this, in which (elements of) several different branches of mathematics are available at the same time. One difficulty in such an enterprise is that the proof-checker may require an unacceptably large level of detail. People refer to the "expansion factor", by which a page of proof written by and for humans expands to ten or more pages of computer-checkable proofs. If the

details could be taken care of automatically, that would advance the subject. On the other hand, some researchers in automated deduction have focused on the attempt to have the computer prove results not previously proved by humans. So far, these efforts have been successful only in areas that can be axiomatized by a few simple axioms, and studied in isolation from the rest of mathematics. We view our work as directed towards opening up wider horizons to automated deduction in the future, not necessarily just as support for proof-checking.

Solving for functions in mathematics

In this section, we will illustrate the theme that "solving for functions" occurs across the board in different branches of mathematics. We have already given the example of Lagrange's theorem in algebra. The context of Lagrange's theorem is typical of a great deal of mathematics: a little set theory, a little number theory, sometimes a little calculus. Two kinds of objects are considered (numbers and elements of the group), along with sets of objects (subgroups), functions from objects to objects (isomorphisms, etc.), and functions from objects to numbers (the order of an element), and even functions from sets of objects to numbers (the order of a group, the index of a subgroup). Objects of greater complexity than that are not required. The recently announced polynomial-time primality test (1), for example, which was hailed in the *New York Times* (Aug. 8, 2002) as the best result in computer science in the past ten years, is actually mathematics of the sort just described. One needs integers, integers mod p, and polynomials over Z_p, groups and their orders, but nothing more complicated. Fermat's conjecture and its long and complex proof notwithstanding, if automated deduction could deal successfully with the simple kinds of contexts just described, the subject would advance rapidly.

We begin with set theory. Consider $S = \{n : P(n)\}$, where P is some property of integers. For simplicity let us just consider sets of integers. The characteristic function χ_S is defined by $\chi_S(n) = 1$ if $n \in S$, 0 if $n \notin S$. What is the relation between χ_S and S? The logician Alonzo Church suggested that in fact the set *is* just its characteristic function. (See the Appendix of (2).) According to Church, any set is a boolean-valued function, and $n \in S$ is just an abbreviation for $\chi_S(n) = \mathbf{true}$. If we take this view, sets are functions, and can be defined by λ-terms. Finding a set with certain desired properties becomes a special case of solving for functions. The "list notation" $\{a, b, c\}$ for a finite set can be regarded as an abbreviation for a function defined by cases: if x is a,b, or c, the value is **true**, else it is **false**.

One can also use tuple notation $\langle x, y\rangle$ and define Cartesian products. Since our interest is not foundational, we ignore the issue of whether and how tuples can be defined in terms of sets. We assume that $\mathbf{p_0}$ and $\mathbf{p_1}$ are pairing functions such that $\langle \mathbf{p_0}z, \mathbf{p_1}z\rangle = z$. We say $A \cong B$ if there is a one-to-one correspondence between A and B. A simple theorem in this subject (just about the simplest set-theoretical theorem I can think of) is that $A \times B \cong B \times A$. ("$\cong$" is read "equipollent"). How do we prove that theorem? By solving for a function f that maps $A \times B$

one-to-one onto $B \times A$. We might like to write that function as $\lambda\langle x, y\rangle.\langle y, x\rangle$, but according to the usual notation for λ-terms, that is not a syntactically correct term. What we want would be formally written as $\lambda z.\langle \mathbf{p_1}(z), \mathbf{p_0}(z)\rangle$. This term arises naturally as the solution to a certain second-order unification problem, but there is no space here for the details.

Turning to number theory, let us consider the relation $n|m$. To define this set (or its characteristic function), we need an existential quantifier: $n|m$ means $\exists k(n * k = m)$. We shall take up this example in a subsequent section, and show that when the existential quantifier is defined using λ-calculus, second-order unification can use this definition to verify that $2|6$ automatically.

Consider a related but more difficult example, the theorem that two positive integers n and m have a greatest common divisor. That is, there exists a number k such that k divides both n and m, and any other integer j that divides both n and m also divides k. We know, from our mathematical education, two different ways to instantiate k: as the least number of the form $\lambda n + \mu m$, where λ and μ are (positive or negative) integers; or as the result of the Euclidean algorithm $E(n, m)$. It seems that neither definition of k will be produced by unification, even though the existence of gcd's follows in first-order logic from the first-order version of Peano's axioms. Is that a slight mystery, since unification is complete for first-order logic? It should not be: Let us abbreviate by $Q(n, m, e)$ the formula that says e is the gcd of n and m. Then there are certain instances of induction (say for simplicity there is just one)

$$I := \forall n(P(n) \to P(n + 1)) \to \forall m P(m)$$

such that in first order logic, $I \wedge Z- > \forall m, n \exists e Q(n, m, e)$. Here Z is the conjunction of the non-induction axioms of arithmetic. Once the proper formula or formulas P is discovered, then resolution driven by unification can, of course, find the proof. Our point here is that finding this proof involves as its main task the discovery of the appropriate instance of induction to use. Now the property P is just a boolean-valued function, and therefore second-order unification can (potentially) find it. We would try to unify the conclusion of the instance of induction, $P(m)$, with the theorem to be proved, in order to find an instantiation of P. Here P would be regarded as a variable (for a function).

We seem to need two sorts or types: numbers and functions. But several sorts (or even infinitely many) can be reduced to first-order, using in this case a unary predicate $N(x)$ for the numbers. We could, if we liked, use another predicate for functions from numbers to booleans, etc.[†]

[†]The distinction between first-order and higher-order logics is not as important as many people assume. Syntactically the two are intertranslatable, using unary predicates to distinguish the types. The difference arises in the *semantics*. The second-order Peano axioms, for example, have an instance of induction for each subset of the integers, of which there are uncountably many. The first-order version of the Peano axioms has induction only for those subsets that can be defined by first-order formulas in a fixed language. Many instances of induction are thus omitted–hence there are non-standard models. There are no non-standard models of the second-order Peano axioms, as Peano himself proved: these axioms characterize

People say that Otter "can't do induction". But induction, with respect to first-order formulae, just involves deduction from first-order axioms, and Otter is as good at that as at any other kind of first-order logic. If the user is willing to specify what instances of induction are required, those instances can be given as axioms to Otter, suitably Skolemized, and Otter can find the proof, at least in some examples I have tried. What Otter cannot do is find the correct instance of induction. This is a problem in "solving for functions", since the instance of induction to be found is a boolean-valued function. In some cases, second-order unification can find the required instance.

One problem here is that we often want to prove a quantified statement by induction. For example in the case of the existence of gcds, the conclusion $Q(n, m, e)$ involves a quantifier. The general plan in automated deduction has been to replace quantifiers by functions. Usually this is done by using symbols for Skolem functions. But if λ-terms are in use, it is possible to deal with quantifiers more directly, using second-order unification to guide the proof. We amplify this point in detail in a subsequent section.

Second-order unification

We focus on the attempt to extend unification to instantiate variables for functions by means of λ-terms. This is loosely known as *higher-order unification*. It has been pursued in the past in the context of formal systems based on typed λ-calculus, so that functions and functionals of any "finite type" can be considered. We call our unification "second-order" since, formally, we prefer a framework without types. This is not essential; our work can be embedded in several different formalisms. In (22), Pietrzykolski gave an algorithm for second-order unification, which we here call λ-unification. This was extended to type theory in (23). Huet showed in (18) that λ-unifiability is more efficient than λ-unification, and subsequently λ-unifiability was used in the implementation of Coq. Huet proved a completeness theorem for λ-unification relative to typed λ-calculus.

λ-unification, and the completeness theorem for it, are for typed λ-calculus without definition by cases. If we try to use λ-unification to find an f such that $f(0) = 0$, it will give us two answers: $f = \lambda x.0$ and $f = \lambda x.x$. If we then ask for an f such that $f(0) = 0$ and $f(1) = 2$, neither of these solutions will do, and indeed no solution is given by a pure λ-term. However, if we look for solutions involving if-then-else, it is easy to define such a function. This example is a fairly representative one, illustrating the two methods "projection" and "imitation" used in the definition of λ-unification.

In (8), a notion of second-order unification is given that allows the construction of functions using definition by cases. Here we call this notion D-unification, to distinguish it from λ-unification. We use $\mathbf{cases}(n, m, x, y)$ to mean "if $n = m$ then x, else y". This unification involves making the "minimal commitment" needed to meet the conditions. Thus, if we use it to solve the problem of finding f such

the integers up to isomorphism. But semantics are not important for automated deduction, which is an inherently syntactic enterprise.

that $f(0) = 0$, the answer we get is $f = \lambda x.\mathbf{cases}(x, 0, 0, Y(x))$, where Y is a fresh variable. Intuitively this means, "if $x = 0$ then 0, else undetermined". Here "undetermined" is different from "undefined", but the idea is similar to "not yet defined". It means that further values of f can be specified by instantiating the new variable Y. This answer is "more general" than the two answers $\lambda x.0$ and $\lambda x.x$ given by Huet's algorithm, in the sense that they can be obtained from it by instantiating Y in different ways. In (8), a most-general-unifier theorem is proved, according to which, if terms t and s (say in typed λ-calculus with definition by cases) unify at all, then they have a unique most general unifier. This is a satisfying generalization of the most-general-unifier property of Robinson's "ordinary" unification.

An important idea in D-unification is that of *restrictions* on a variable. A restriction is a pair consisting of a variable and a (possibly empty) list of constants. Note that if a problem expressed in first-order logic using quantifiers is converted to clausal form, some of the originally-bound variables are converted to constants, so the concept "constant" here includes what sometimes are called "object variables", and the concept "variable" here includes what is sometimes called "metavariable". The idea is that the list of constants paired with x are *forbidden to* x.[‡] Unification is not allowed to assign a variable x a value that contains a constant forbidden to x. To make this sensible, the input to the unification algorithm has to include an *environment*, which is a finite list of restrictions. Since D-unification can introduce new variables, not mentioned in the input environment, the output of the unification algorithm is not only a substitution but also an *output environment*. The substitution σ unifies t and s relative to environment E if for some substitution χ whose restriction to E is the identity, we have $t\sigma\chi = s\sigma\chi$. That is, it is possible to extend σ to give the new variables values such that, under the extended substitution, t and s become equal.

We will state the most-general-unifier theorem, which is the nicest property of D-unification. Before the theorem can be stated, the notion of "most general substitution" must be defined. Here are the definition and the theorem, taken from (8).

Definition: Given an environment E, θ is more general than μ, relative to E, if there is a substitution β such that $\theta\beta = \mu$ on E. That is, for all variables X in the environment E, we have $X\theta\beta \cong X\mu$. Here $p \cong q$ means $px = qx$, where x is a variable not in p or q.

Remark. In this definition, the phrase "$px = qx$" means that $px = qx$ is provable in the λ-calculus with definition by cases (be it a typed version or the untyped theory λ-D considered in (8)).

[‡]Technically, one has to decide whether x in $\lambda x.t$ is a variable or a constant. Conceptually it is constant, since unification cannot assign it a value. But if one takes that seriously, one must constantly be replacing variables by constants and vice-versa when β-reduction removes λ. In the Otter implementation we do not do this.

THEOREM 0.1: [Most general unifier] *Let E be an environment and p and q normal terms. Suppose that for some substitution θ legal for E, $p\theta$ and $q\theta$ are identical. Then D-unification terminates successfully on inputs E,p, and q, and the answer substitution is legal for E, and more general than θ.*

Comparison of D-unification and λ-unification

The uniqueness of the result of D-unification, and the most-general-unifier theorem, may seem puzzling in view of the many-valuedness of λ-unification. As the author of (8), I received a number of inquiries about this point. Is λ-unification many-valued only because of examples like the one above (in which it seems that the reason is over-specification of the unifier), or also for other, possibly more fundamental reasons? How do the two theorems (completeness of Huet's algorithm and most-general-unifier theorem) avoid contradicting each other? The purpose of this section is to answer these questions.

Let us consider the statement of the theorems carefully. The hypothesis of the most-general-unifier theorem of (8) is that t and s are given normal terms, and there is a substitution θ such that $t\theta$ and $s\theta$ are identical. In that case, says the theorem, there is a most general unifier. The terms t and s, as well as the unifier, can involve definition by cases. What if we replace the hypothesis that $t\theta$ and $s\theta$ are identical by the weaker requirement that $t\theta$ and $s\theta$ are provably equal (i.e., in view of Church-Rosser, they have the same normal form)? Is the theorem still true? Also, if t and s are provably equivalent in λd-calculus, are they necessarily unifiable? These questions are not answered in (8).

The completeness theorem for λ-unification has the weaker hypothesis, that $t\theta$ and $s\theta$ have the same normal form, but neither the terms t and s nor θ can involve **cases**. The conclusion is that there is a complete set of unifiers (a CSU) for t and s, i.e. every unifier is more general than some substitution in the CSU. The CSU can be infinite.

It will be instructive to consider an example from (22), which was also considered in (18), and compare how the example is treated by λ-unification and D-unification. The example is the unification of $F(F(X))$ with $a(a(b))$. Here F and X are variables and a and b are constants. **cases**-unification finds (only) the solution $\{F := a, X := b\}$. λ-unification finds this solution, and also the solution $\{F := \lambda u.a(a(b)), X := b\}$ (here F is a constant function) and the solution $\{F := \lambda u.u, X := a(b)\}$ (here F is the identity function). The details of the calculation for λ-unification are on p. 42 of (18).[§] The example brings to the fore the fact that D-unification has no clause in its definition that applies to the case of $F(t)$, where t is a compound term containing variables forbidden to F, such as (in this case) F itself. Therefore only the "Robinson clause" (which is the same as first-order unification) applies to this example, which is why we get only the first solution.

[§]There is a technicality about whether η-reduction is used in λ-unification or not. It is used in (22). If it is not used, as in (18), we get one more solution, $\{F := \lambda u.a(u), X := b\}$, which is η-equivalent to $\{F := a, X := b\}$.

This example shows that the most general unifier theorem for D-unification depends critically on the hypothesis that $t\sigma$ and $s\sigma$ are identical, not just β-convertible. To see this, take t to be $F(F(X))$ and s to be $a(a(b))$. Take σ to be the second solution substitution above. Then $t\sigma$ and $s\sigma$ are beta-convertible, but it is not the case that σ is more general than $\{F := a, X := b\}$; indeed since this substitution has constants on the right, it is not more general than any other substitution.

An even simpler example can be given to show that the most general unifier theorem fails if we change the hypothesis to "$t\sigma$ and $s\sigma$ are β-convertible" instead of "identical". Consider the unification problem $X(t) = t$, where t is a compound term. Then D-unification does not succeed, as there is no clause in the definition that applies. But if we take θ to be the substitution $\{X := \lambda x.x\}$, then $X(t)\theta$ is β-convertible to $t\theta$. This would be a counterexample to the theorem with the changed hypothesis.

Implementation in Otter

D-unification has earlier been implemented in a backwards-Genzten theorem prover (9). However, that prover is not as strong or robust as Otter, and the well-known powers of Otter should work well in combination with D-unification. Implementation of this algorithm in the source code of the theorem-prover Otter is well under way at the time of writing (August 2002). There are some additional factors: Otter already has a large user community, so implementation in Otter will make D-unification readily available, without anyone having to learn a new system. Also, Fitelson and Harris have written a Mathematica interface to Otter (private communication), which can be used to connect *Theorema* to Otter.

The implementation had to begin with adding λ-calculus to Otter. Reserved words `lambda` and `ap` (or synonymously `Ap`) are used for this purpose. One uses `lambda(x,ap(f,x))` to enter the term $\lambda x.f(x)$. Beta-reduction has been implemented in the framework Otter uses for demodulation. Technicalities necessary to avoid clash of bound variables have been successfully dealt with. The following Otter proof shows a beta-reduction combined with an ordinary demodulation, given the demodulator $x * x = x$. The theorem proved is

$$(\lambda x.x * x)c = c.$$

Of course the proof is trivial: it is only meant to demonstrate the successful implementation of β-reduction working more or less the same way as ordinary demodulation in a first-order theorem-prover. Line 3 of the proof is the negation of the goal. The first two lines are axioms. The proof completes with the derivation of a contradiction.

```
1 [] x=x.
2 [] x*x=x.
3 [] ap(ap(lambda(x,lambda(y,x)),c*c),y)!=c.
4 [3,demod,2,beta,beta] c!=c.
5 [binary,4.1,1.1] .
```

Quantification in a clausal theorem-prover

In this section we show how to treat quantification in Otter, at the clause level, based on the machinery of λ-calculus and second-order unification. An example proof will be worked through in detail.

One can regard $\exists$ as a boolean-valued functional, whose arguments are boolean functions (defined, for simplicity, on the integers, let's say). Then $\exists n.P(n)$ is an abbreviation for $\exists(\lambda n.P(n))$, or more explicitly, $Ap(\exists, \lambda n.P(n))$. It is then possible to work with quantified statements directly in Otter–that is, in the version of Otter that is enhanced with λ-calculus.

We will show exactly how this is done. In the presence of λ-calculus, $\exists$ is treated as a constant. The rule for dealing with $\exists$ in Otter is

```
-Ap(Z,w) | exists(lambda(x, Ap(Z,x))
```

This works in Otter as follows: if $Z(t)$ can be proved for any term t, then the literal $-Ap(Z, x)$ will be resolved away, using the substitution $x := t$. Then $\exists(\lambda x.Z(x))$ is deduced, which can be abbreviated to $\exists x.Z(x)$. No machinery is added to Otter to accomplish this, other than what has already been added for λ-calculus.

This will permit the use of definitions that explicitly involve a quantified formula in the definition. For example, we could define

```
divides(u,v) = exists(lambda(x,u*x = v)).
```

Now, given the clause $2 * 3 = 6$, Otter can deduce `divides(2,6)` as follows: First, the negated goal `-divides(2,6)` will rewrite to

`-exists(lambda(x,2*x = 6))`.

This will unify with `exists(lambda(x,Ap(Z,x))` if `2*x=6` will unify with unify with `Ap(Z,x)`. Second-order unification (with x forbidden to Z) will find

$$Z = \lambda w.(2 * w = 6 \vee Y(w))$$

where Y is a new variable. Resolution of the two clauses containing `exists` will then generate a new clause containing the single literal $-Ap(Z, w)$ with this value of Z. Specifically,

$$-Ap(\lambda w.(2 * w = 6 \vee Y(w)), w).$$

That will be β-reduced to $-(2 * w = 6 \vee Y(w))$ so the clause finally generated will be `-or(2*w = 6, Y(w))`. Demodulation can apply to a negated literal under these circumstances and produce two new clauses, `-(2*w = 6)` and $-Y(w)$.¶ The first of these resolves with $2 * 3 = 6$, producing a contradiction that completes the proof.

¶Demodulation technically leads from term to term, or clause to clause. We refer here to a generalized version of demodulation which treats certain functors, such as `or`, specially. It also incorporates β-reduction.

Two simple examples

The first example is the following theorem: If $a \neq b$, there exists a predicate which is true on a and false on b. The input file contains some equations for **cases** as well as the clauses `a != b` and `-Ap(X,a) | Ap(X,b)`, which is the way we write $\neg X(z) \vee X(b)$ in Otter. The inference rules are binary resolution and paramodulation. The binary resolution rule has been modified so that a term $X(a)$ or its negation can always be unified with the constant `$F`, which is Otter's name for falsity. Unifying `-Ap(X,a)` with `$F` Otter finds `X = lambda(x,cases(x,a,$T,Ap(Y,x))` , where `Y` is a new variable. The result of the binary resolution step is `Ap(lambda(x,cases(x,a,$T,Ap(Y,x)),b)`. This beta-reduces to `cases(b,a,$T,Ap(Y,b))`. But before this, Otter has already deduced `cases(b,a,x,y) = y`. Since we have set the option `knuth-bendix`, which turns on paramodulation, this previously-deduced equation is used as a demodulator, and our new clause demodulates to `Ap(Y,b)`. This clause then unifies with `$F`, producing `Y = lambda(y,cases(y,b,$F,Ap(Z,y))`. The resolvent is the empty clause, completing the proof. The value of the function found is, after beta-reduction, `X = lambda(x,cases(x,a,$T,cases(x,b,$F, Ap(Z,x))))`. Intuitively, this function is true on a, false on b, and elsewhere undetermined.

Our second example is to prove the existence of the identity function. The goal is $\exists F \forall x(x = Ap(F, x))$. Traditionally, to formulate a problem of the form $\exists x \forall y P(x, y)$ for Otter, one would introduce a Skolem function g, and write the negated goal as $\neg P(x, g(x))$. By comparison, in a backwards Gentzen prover (such as was used in (9)), one would change y to a constant forbidden to x, and the negated goal (which one does not have to negate in such a prover, but we negate it for comparison to Otter) would be $\neg P(x, c)$, with c forbidden to x. Intuitively, these two goals are similar, since $g(x)$ functions more or less the same as a constant forbidden to x. It is like an arbitrary constant because the Skolem function is not further specified–$g(x)$ could be any arbitrary value–and it is "forbidden to x" by the occurs check in unification.

We take the negated goal in the form $\neg P(x, c)$, with c forbidden to x. In our example this is $c \neq Ap(F, c)$. We resolve this with the equality axiom $x = x$. First x is given the value c. Then we have to unify $Ap(F, c)$ with c, where c is forbidden to F. The definition of D-unification tells us $F := \lambda x.(x \vee Yx)$, where Y is a new variable. The succesful unification derives a contradiction by binary resolution and completes the proof of the theorem.

On the other hand, if we take the Skolemized version of the negated goal, we have to unify $Ap(F, g(F))$ with $g(F)$, where g is a Skolem function. We have discussed this shortcoming (being unable to unify $X(t)$ with t) in an earlier section. Until this difficulty is overcome, we use constants forbidden to F when converting a theorem to clausal form.

A more mathematical example: Lagrange's theorem

In this section we use D-unification to work out the most important details of the proof of the algebraic part of Lagrange's theorem: the existence of a one-to-one correspondence between H and the coset Ha, where $a \in G$.

To prepare this for Otter, we use a unary predicate $G(x)$ for the group G, a unary predicate $H(x)$ for the subgroup H, and include the axioms that assert that G is a group under $*$ with identity e and inverse $i(x)$, and that H is closed under $*$ and inverse, and $H(x)$ implies $G(x)$. We also include the following:

```
 -Ap(Z,w) | exists(lambda(x,Ap(Z,x))).
G(a).
```

Now consider how to formalize the coset Ha. We have

$$\begin{aligned} Ha &= \{ha | h \in H\} \\ &= \{w : \exists h.(h \in H \wedge h * a = w\} \\ &= \lambda w.(\exists h.(h \in H \wedge h * a = w) \\ &= \lambda w.(\exists(\lambda(h, H(h) \wedge h * a = w))) \end{aligned}$$

We put this into Otter using a function symbol `ha`, as follows:

```
ha(w) = exists(lambda(h, and(H(h), h*a = w))).
```

We use this as a demodulator, so that it will be used to rewrite any literal -ha(t) that arises. Such a literal would then resolve with existential axiom, using D-unification, and the resulting substitution would yield

```
Z:= lambda(h, or(and(H(h),h*a = t), Ap(Y,h)))
```

where Y is a new variable.

Now, for simplicity, we begin by proving that there is a function F from H to Ha, without worrying about the one-to-one and onto part yet. The goal is then

$$\exists F \forall x(x \in H \rightarrow Ap(F, x) \in Ha)$$

Instead of Skolemizing $x = g(F)$, we replace x by a constant that is forbidden to F. The negated goal becomes the two clauses

```
H(c).
-ha(Ap(F,c)).
```

As shown above, the literal `-ha(Ap(F,c))` demodulates and resolves with the existential axiom; the unification produces the substitution

```
Z:= lambda(h, or(and(H(h),h*a = Ap(F,c)), Ap(Y,h)))
```

and the resolution produces the unit clause

```
-Ap(lambda(h,or(and(H(h),h*a = Ap(F,c)), Ap(Y,h))),w).
```

This clause however is not stored yet, because it β-reduces to

```
-or(and(H(w),w*a = Ap(F,c)),Ap(Y,w))).
```

Demodulators are used to implement de Morgan's laws, so this will demodulate to the two clauses `-H(w) | w*a != Ap(F,c)` and `-Ap(Y,w)`. Now we're getting somewhere! The literal `-H(w)` resolves with `H(c)`. The inferred clause is $c * a! = Ap(F, c)$. The single literal in this clause can be unified with the equality axiom $x = x$. This results in unifying $c * a$ with $Ap(F, c)$. Since c is forbidden to F, the answer substitution is $F := \lambda x.x * a \vee Y(x)$. If we take $Y = \lambda x.\mathbf{false}$ we have $F = \lambda x.x * a$ (since $u \vee \mathbf{false} = u$, even if u is not Boolean). $\lambda x.x * a$ is the desired map from H to the coset Ha. The proof that it is one-to-one and onto is relatively straightforward.

Related work

The pioneer of the actual use of second-order unification was Huet (18). It has been implemented in Coq, which is described in (3) as well as numerous documents available from the Coq web page. There are many existing implementations of higher-order logic, including: λ-prolog (20); PVS, which is being used at SRI under the direction of N. Shankar (21); HOL-Light, which was written by John Harrison (16; 17) and is currently being used by him at Intel's Portland facility; NuPrl, developed at Cornell under the direction of Constable (14); and the French system Coq developed at INRIA, which is also being used in Nijmegen. These systems (if they implement higher-order unification at all) use λ-unification, rather than D-unification, and they are primarily proof-checkers, not proof-finders. We believe that D-unification offers greater power and efficiency, as does the use of the industrial-strength clausal theorem prover Otter. A proof of Lagrange's theorem has been checked by the use of Nqthm (25).

Some years ago, Quaife used Otter to formalize set theory using a finite axiomatization due in essence to Gödel-Bernays. He then used set theory to formulate Peano arithmetic, and he succeeded in proving the fundamental theorem of arithmetic (24). Belinfante, building on Quaife's beginning, has carried forward the formal development of set theory from first principles, proving thousands of theorems in Otter (10; 11); see also Belinfante's web site.

References

1. Agrawal, M., Saxena, N., and Kayal, N., PRIME is in P, available from `http://www.cse.iitk.ac.in/news/primality.html`

2. Barendregt, H., *The Lambda Calculus: Its Syntax and Semantics*, Studies in Logic and the Foundations of Mathematics **103**, Elsevier Science Ltd. Revised edition (October 1984).

3. Barendregt, H., and Geuvers, H., Proof-Assistants Using Dependent Type Systems, in: Robinson, A., and Voronkov, A. (eds.), *Handbook of Automated Reasoning, vol. II*, pp. 1151-1238. Elsevier Science (2001).

4. Beeson, M., Some applications of Gentzen's proof theory to automated deduction, in P. Schroeder-Heister (ed.), *Extensions of Logic Programming*, Lecture Notes in Computer Science **475** 101-156, Springer-Verlag (1991).

5. Beeson, M., *Mathpert*: Computer support for learning algebra, trigonometry, and calculus, in: A. Voronkov (ed.), Logic Programming and Automated Reasoning, Lecture Notes in Artificial Intelligence 624, Springer-Verlag (1992).

6. Beeson, M., *Mathpert Calculus Assistant*. This software product was published in July, 1997 by Mathpert Systems, Santa Clara, CA. See www.mathxpert.com to download a trial copy.

7. Beeson, M., Automatic generation of epsilon-delta proofs of continuity, in: Calmet, Jacques, and Plaza, Jan (eds.) *Artificial Intelligence and Symbolic Computation: International Conference AISC-98, Plattsburgh, New York, USA, September 1998 Proceedings*, pp. 67-83. Springer-Verlag (1998).

8. Beeson, M., Unification in Lambda Calculus with if-then-else, in: Kirchner, C., and Kirchner, H. (eds.), *Automated Deduction-CADE-15. 15th International Conference on Automated Deduction, Lindau, Germany, July 1998 Proceedings*, pp. 96-111, Lecture Notes in Artificial Intelligence **1421**, Springer-Verlag (1998).

9. Beeson, M., A second-order theorem prover applied to circumscription, in: Gor, R., Leitsch, A., and Nipkow, T. (eds.), *Automated Reasoning, First International Joint Conference, IJCAR 2001, Siena, Italy, June 2001, Proceedings*, Lecture Notes in Artificial Intelligence **2083**, Springer-Verlag (2001).

10. Belinfante, J., Computer proofs in Gödel's class theory with equational definitions for composite and cross, *J. Automated Reasoning* **22**, No. 3 (1988), pp. 311-339.

11. Belinfante, J., On computer-assisted proofs in ordinal number theory, *J. Automated Reasoning* **22**, No. 3, pp. 341-378.

12. Boyer, R. S., and Moore, J. S., *A Computational Logic Handbook*, Academic Press, Boston (1988).

13. Buchberger, B., *et. al.* Theorema: An Integrated System for Computation and Deduction in Natural Style, in: *Proceedings of the Workshop on Integration of Deductive Systems at CADE-15, Lindau, Germany, July 1998*

14. Constable, R. L. *et. al.*, *Implementing Mathematics with the Nuprl Proof Development System*, Prentice-Hall, Englewood Cliffs, New Jersey (1986).

15. Hall, Marshall, Jr., *The Theory of Groups*, Macmillan, New York (1959).

16. Harrison, J., and Théry, L.: Extending the HOL theorem prover with a computer algebra system to reason about the reals, in *Higher Order Logic Theorem Proving and its Applications: 6th International Workshop, HUG '93*, pp. 174–184, Lecture Notes in Computer Science **780**, Springer-Verlag (1993).

17. Harrison, J., *Theorem Proving with the Real Numbers*, Springer-Verlag, Berlin/Heidelberg/New York (1998).

18. G. Huet. A unification algorithm for typed λ-calculus. *Theoretical Computer Science* **1** 27–52, 1975.

19. McCune, W.: Otter 2.0, in: Stickel, M. E. (ed.), *10th International Conference on Automated Deduction* pp. 663–664, Springer-Verlag, Berlin/Heidelberg (1990).

20. D. Miller and G. Nadathur. An Overview of λ-Prolog In *Proceedings of the Fifth International Symposium on Logic Programming, Seattle, August 1988*.

21. Owre, S., Rushby, J. M., Shankar, N., PVS: A Prototype Verification System, in: Kapur, D. (ed.), *Automated Deduction–CADE-11, 11th International Conference on Automated Deduction*, 748–752, LNCS **607**, Springer-Verlag (1992).

22. Pietrzykowski, T., and Jensen, D., A complete mechanization of second order logic, *J. Assoc. Comp. Mach.* **20** (2) pp. 333-364, 1971.

23. Pietrzykowski, T., and Jensen, D., A complete mechanization of ω-order type theory, *ASsoc. Comp. Math. Nat. Conf.* 1972, Vol. 1, 82–92.

24. Quaife, A., *Automated Development of Fundamental Mathematical Theories, Automated Reasoning, Vol.2*, Kluwer Academic Publishers, Dordrecht (1992).

25. Yuan Yu, Computer proofs in group theory, *J. Automated Reasoning* **6**(3) pp. 251–286, 1990.

Computing restrictions of ideals in finitely generated k-algebras by means of Buchberger's algorithm

Thomas Beth, Jörn Müller-Quade and
Rainer Steinwandt

Institut für Algorithmen und Kognitive Systeme, Fakultät für Informatik,
Am Fasanengarten 5, Universität Karlsruhe, 76128 Karlsruhe, Germany

Abstract

Gröbner bases can be used to solve various algorithmic problems in the context of finitely generated field extensions. One key idea is the computation of a certain kind of restriction of an ideal to a subring. This contribution generalizes this approach to allow the computation of the restriction of an arbitrary ideal to a subring.

It is tempting to hope that this technique can be used to compute a generating set for the intersection of finitely generated extension fields; in fact, in (Müller-Quade Beth 1998*a*) such an algorithm has been sketched. Here we give a (counter-)example which shows that this algorithm does not work in general. Even though we show that Gröbner bases can be used to compute a significantly more general kind of ideal restrictions than considered in the context of finitely generated field extensions, the application of these methods to the field intersection problem remains an interesting open problem.

KEYWORDS: Gröbner bases, finitely generated function fields

1. Introduction

Buchberger's algorithm allows in particular for a constructive theory of ideals in polynomial rings, that led to a multitude of applications to which parts of this conference is devoted. Many computational problems, especially concerning finitely generated field extensions, can be solved by the restriction of specific ideals to rings defined over a subfield. This paper presents a novel algorithm which allows to restrict an *arbitrary* ideal $\mathfrak{I} = \langle f_1, \ldots, f_s \rangle$ in a residue class ring

The material in this paper was presented in part at the Seventh Rhine Workshop on Computer Algebra (Steinwandt Müller-Quade 2000).

$k(\vec{x})[Z_1,\ldots,Z_n]/\langle\vec{q}\rangle$ to a subring $k(\vec{g})[Z_1,\ldots,Z_n]/\langle\vec{q}\rangle$ defined over a subfield $k(\vec{g})$ of $k(\vec{x})$, where $\vec{q}\in k(\vec{g})[Z_1,\ldots,Z_n]$. So for the special case $\vec{q}=\vec{0}$ we are dealing with a question on subalgebras of the polynomial ring $k(\vec{x})[Z_1,\ldots,Z_n]$. However, in difference to (Kapur Madlener 1989, Robbiano Sweedler 1990), for instance, here we do not focus on $k(\vec{x})$-subalgebras of the form $k(\vec{x})[a_1,\ldots,a_r]$ with $a_1,\ldots,a_r\in k(\vec{x})[Z_1,\ldots,Z_n]$; instead we are interested in k-subalgebras $k(\vec{g})[Z_1,\ldots,Z_n]$ that are obtained by restricting the field of coefficients $k(\vec{x})$.

Given finitely many generators $f_1,\ldots,f_s$ for an ideal $\mathfrak{I}\subseteq k(\vec{x})[Z_1,\ldots,Z_n]/\langle\vec{q}\rangle$, our algorithm computes generators for the ideal $\mathfrak{I}\cap k(\vec{g})[Z_1,\ldots,Z_n]/\langle\vec{q}\rangle$; here we identify $k(\vec{g})[Z_1,\ldots,Z_n]/\langle\vec{q}\rangle$ with its image under the natural embedding of $k(\vec{g})[Z_1,\ldots,Z_n]/\langle\vec{q}\rangle$ into $k(\vec{x})[Z_1,\ldots,Z_n]/\langle\vec{q}\rangle$.

An instructive example are minimal polynomials: if one restricts the ideal $\langle Z-\alpha\rangle\subseteq k(\alpha)[Z]$ to $k[Z]$ then $\langle Z-\alpha\rangle\cap k[Z]$ is a principal ideal and a monic generator of this ideal is the minimal polynomial of α over k.

This work is motivated by the implications of Buchberger's algorithm to finitely generated field extensions. Apart from the so-called tag variable approach (Sweedler 1993, Kemper 1993) many problems concerning field extensions can be solved by means of ideal restriction. One can employ a correspondence between the lattice of subfields $k(\vec{g})$ of a finitely generated field $k(\vec{x})$ and a lattice of restricted ideals $\mathfrak{P}_{(\vec{x})/k(\vec{g})}:=\langle Z_1-x_1,\ldots,Z_n-x_n\rangle\cap k(\vec{g})[Z_1,\ldots,Z_n]$. This correspondence allows to solve many problems concerning field extensions by means of constructive ideal theory and Buchberger's algorithm (Müller-Quade Steinwandt 1999, 2000). Many characteristic properties of subfields directly translate to properties of the restricted ideals. E. g., the transcendence degree of the extension $k(\vec{g})\leq k(\vec{x})$ equals the dimension of the ideal $\mathfrak{P}_{(\vec{x})/k(\vec{g})}$, the polynomial $n(Z_1,\ldots,Z_n)-\frac{n(\vec{x})}{d(\vec{x})}d(Z_1,\ldots,Z_n)$ reduces to zero modulo a Gröbner basis of $\mathfrak{P}_{(\vec{x})/k(\vec{g})}$ iff $\frac{n(\vec{x})}{d(\vec{x})}$ is contained in $k(\vec{g})$, field extensions correspond to ideal inclusions, and the coefficients of a reduced Gröbner basis of $\mathfrak{P}_{(\vec{x})/k(\vec{g})}$ yield a canonical generating set of the field $k(\vec{g})$.

For the specific ideals used in the above correspondence the restriction problem was solved in (Müller-Quade Steinwandt 1999, 2000). But the more general question of restricting an arbitrary ideal $\mathfrak{I}$ to a finitely generated subfield was posed in (Müller-Quade Beth 1998*a*). In an approach to solve the field intersection problem the constructive restriction of more general ideals was used as a subroutine. The purpose of this paper is twofold. First we will solve the general ideal restriction problem and second we will show that the algorithm of (Müller-Quade Beth 1998*a*) cannot in all cases calculate the intersection of two finitely generated fields.

Even though the general ideal restriction problem can be solved by Gröbner basis techniques, including computing the field of definition of an ideal, ideal saturation, primary decomposition, and ideal membership, the general field intersection problem remains an interesting open problem to solve.

2. Restricting ideals in finitely generated k-algebras

To avoid ambiguities, we start by summarizing the notation that we use in the sequel:

- For K a field we write $K[\vec{Z}] := K[Z_1, \ldots, Z_n]$ for the polynomial ring in the indeterminates $Z_1, \ldots, Z_n$ over the field (of coefficients) K.
- For K a field, $\mathfrak{I} \subseteq K[\vec{Z}]$ an ideal, and $q \in K[\vec{Z}]$ we denote by $\mathfrak{I} : q^\infty$ the saturation of $\mathfrak{I}$ w. r. t. q, i. e.,

$$\mathfrak{I} : q^\infty = \{p \in K[\vec{Z}] : q^\mu \cdot p \in \mathfrak{I} \text{ for some } \mu \in \mathbb{N}\}.$$

- For indeterminates $X_1, \ldots, X_u$ we denote by $\mathrm{T}(\vec{X})$ the set of terms in $\vec{X}$, i. e., the set of all products $\prod_{i=1}^{u} {X_i}^{\nu_i}$ with $\nu_1, \ldots, \nu_u \in \mathbb{N}_0$.
- $k(\vec{x}) := k(x_1, \ldots, x_n)$ denotes a finitely generated extension field of some ground field k. We assume computations in $k(\vec{x})$ to be effective and that $k(\vec{x})$ is represented as the quotient field of $k[X_1, \ldots, X_n]/\langle b_1 \ldots, b_t\rangle$ where $\vec{b}$ is a finite system of generators of the 'ideal of relations'

$$\mathfrak{P}_{(\vec{x})/k} := \left\{ a(\vec{X}) \in k[\vec{X}] : a(\vec{x}) = 0 \right\}.$$

- $k(\vec{g}) := k(g_1, \ldots, g_r)$ denotes a subfield of $k(\vec{x})$ generated over the ground field k by $g_1, \ldots, g_r \in k(\vec{x})$.
- For an ideal $\mathfrak{I} \subseteq k(\vec{x})[\vec{Z}]$ we denote by $k_\mathfrak{I}$ the minimal field of definition of $\mathfrak{I}$. In other words, $k_\mathfrak{I}$ is the field generated over the prime field of k by the coefficients occurring in a reduced Gröbner basis of $\mathfrak{I}$ (cf. (Müller-Quade Rötteler 1998, Robbiano Sweedler 1998)).
- $\mathfrak{Q} := \langle q_1, \ldots, q_v\rangle \in k(\vec{x})[\vec{Z}]$ denotes an ideal whose minimal field of definition $k_\mathfrak{Q}$ is contained in $k(\vec{g})$. As by computing a reduced Gröbner basis of $\mathfrak{Q}$ a generating set $\mathcal{B} \subseteq k_\mathfrak{Q}[\vec{Z}]$ of $\mathfrak{Q}$ can be derived, we assume w. l. o. g. $q_1, \ldots, q_v \in k_\mathfrak{Q}[\vec{Z}] \subseteq k(\vec{g})[\vec{Z}]$.
- By $\pi_g : k(\vec{g})[\vec{Z}] \longrightarrow k(\vec{g})[\vec{Z}]/\langle\vec{q}\rangle$ and $\pi_x : k(\vec{x})[\vec{Z}] \longrightarrow k(\vec{x})[\vec{Z}]/\langle\vec{q}\rangle$ we denote the canonical residue class epimorphisms.
- $\iota : k(\vec{g})[\vec{Z}]/\langle\vec{q}\rangle \longrightarrow k(\vec{x})[\vec{Z}]/\langle\vec{q}\rangle$ denotes the natural $k(\vec{g})$-algebra monomorphism that maps $\pi_g(Z_i)$ to $\pi_x(Z_i)$ $(i = 1, \ldots, n)$. In particular, we can identify $k(\vec{g})[\vec{Z}]/\langle\vec{q}\rangle$ with the subring $\iota(k(\vec{g})[\vec{Z}]/\langle\vec{q}\rangle)$ of $k(\vec{x})[\vec{Z}]/\langle\vec{q}\rangle$.

With this notation we can summarize the computational task to be solved in this section as follows:

> *Given a finite basis $f_1, \ldots, f_s$ of an ideal $\mathfrak{I} = \langle f_1, \ldots, f_s\rangle \subseteq k(\vec{x})[\vec{Z}]/\langle\vec{q}\rangle$, compute a finite generating set of the restricted ideal $\mathfrak{I} \cap \iota(k(\vec{g})[\vec{Z}]/\langle\vec{q}\rangle)$.*

To compute this restriction, in a first step we determine a finite basis $\vec{p} \subseteq k(\vec{g})[\vec{X}]$ of the ideal

$$\mathfrak{P}_{(\vec{x})/k(\vec{g})} := \Big\{ a(\vec{X}) \in k(\vec{g})[\vec{X}] : a(\vec{x}) = 0 \Big\} = \langle Z_1 - x_1, \ldots, Z_n - x_n \rangle \cap k(\vec{g})[\vec{Z}].$$

This task can be accomplished by means of the following result (see (Müller-Quade et al. 1998, Proposition 1)):

LEMMA 2.1: *With the above notation let $g_i = n_i(\vec{x})/d_i(\vec{x})$ with $n_i, d_i \in k[\vec{X}]$ and $d_i(\vec{x}) \neq 0$ $(1 \leq i \leq r)$. Then*

$$\mathfrak{P}_{(\vec{x})/k(\vec{g})} = \Big\langle n_1(\vec{X}) - g_1 \cdot d_1(\vec{X}), \ldots, n_r(\vec{X}) - g_1 \cdot d_r(\vec{X}), b_1, \ldots, b_t \Big\rangle : \Big(\prod_{i=1}^{r} d_i(\vec{X}) \Big)^{\infty}.$$

For effectively computing the saturation in Lemma 2.1 we can apply (Becker Weispfenning 1993, Proposition 6.37), for instance. Next, we fix for each generator f_i of the ideal $\mathfrak{I}$ a representative $f_i(\vec{X}, \vec{Z}) \in k(\vec{X})[\vec{Z}]$, i. e., for $i = 1, \ldots, s$ we have $f_i = \pi_x(f_i(\vec{x}, \vec{Z}))$. By 'clearing denominators' we may select polynomials $\tilde{d}_i(\vec{X}) \in k[\vec{X}]$ such that for $i = 1, \ldots, s$ both $\tilde{d}_i(\vec{x}) \neq 0$ and

$$F_i = F_i(\vec{X}, \vec{Z}) := \tilde{d}_i(\vec{X}) \cdot f_i(\vec{X}, \vec{Z}) \in k[\vec{X}, \vec{Z}]$$

hold. Now the essential tool we will use both for characterizing and for computing the restricted ideal $\mathfrak{I} \cap \iota(k(\vec{g})[\vec{Z}]/\langle \vec{q} \rangle)$ is the ideal

$$\mathfrak{H} := \sum_{h \in k(\vec{g})[\vec{X}] \setminus \mathfrak{P}_{(\vec{x})/k(\vec{g})}} \Big(\langle \vec{F}, \vec{p}, \vec{q} \rangle : h^{\infty} \Big) \subseteq k(\vec{g})[\vec{X}, \vec{Z}].$$

Exploiting the fact that $k(\vec{g})[\vec{X}, \vec{Z}]$ is noetherian, it is not difficult to see that $\mathfrak{H}$ can in fact be written as a simple saturation:

Remark: With the above notation, there is a polynomial $h_0 \in k(\vec{g})[\vec{X}] \setminus \mathfrak{P}_{(\vec{x})/k(\vec{g})}$ such that $\mathfrak{H} = \langle \vec{F}, \vec{p}, \vec{q} \rangle : {h_0}^{\infty}$.

Proof: As $k(\vec{g})[\vec{X}, \vec{Z}]$ is noetherian, there is a finite subset $\mathcal{P} \subseteq k(\vec{g})[\vec{X}]$ with $\mathfrak{H} = \sum_{h \in \mathcal{P}} \Big(\langle \vec{F}, \vec{p}, \vec{q} \rangle : h^{\infty} \Big)$. For a finite sum one easily checks the inclusion

$$\sum_{h \in \mathcal{P}} \Big(\langle \vec{F}, \vec{p}, \vec{q} \rangle : h^{\infty} \Big) \subseteq \langle \vec{F}, \vec{p}, \vec{q} \rangle : \left(\prod_{h \in \mathcal{P}} h \right)^{\infty}. \tag{1}$$

Thus, setting $h_0 := \prod_{h \in \mathcal{P}} h \in k(\vec{g})[\vec{X}]$, we have $h_0 \notin \mathfrak{P}_{(\vec{x})/k(\vec{g})}$, because of the latter being a prime ideal. Moreover, from Equation (1), we also know that $\mathfrak{H} \subseteq \langle \vec{F}, \vec{p}, \vec{q} \rangle : {h_0}^{\infty}$. Equality follows from $h_0 \in k(\vec{g})[\vec{X}] \setminus \mathfrak{P}_{(\vec{x})/k(\vec{g})}$, i. e., h_0 is one of the summands occurring in the defining sum of $\mathfrak{H}$. □

By means of the ideal $\mathfrak{H}$, the restricted ideal $\mathfrak{I} \cap \iota(k(\vec{g})[\vec{Z}]/\langle\vec{q}\rangle)$ can now be characterized as follows:

LEMMA 2.2: *With the above notation we have*

$$\mathfrak{I} \cap \iota(k(\vec{g})[\vec{Z}]/\langle\vec{q}\rangle) = (\iota \circ \pi_g)(\mathfrak{H} \cap k(\vec{g})[\vec{Z}]).$$

Proof: '$\supseteq$': From the above remark we know that there exists a polynomial $h_0 \in k(\vec{g})[\vec{X}] \setminus \mathfrak{P}_{(\vec{x})/k(\vec{g})}$ with $\mathfrak{H} = \langle\vec{F}, \vec{p}, \vec{q}\rangle : {h_0}^{\infty}$.

Now let $a \in (\iota \circ \pi_g)(\mathfrak{H} \cap k(\vec{g})[\vec{Z}])$ and $a(\vec{Z}) \in (\iota \circ \pi_g)^{-1}(a)$, i. e., for a suitable $\mu \in \mathbb{N}$ we have ${h_0}^{\mu} \cdot a(\vec{Z}) \in \langle\vec{F}, \vec{p}, \vec{q}\rangle \subseteq k(\vec{g})[\vec{X}, \vec{Z}]$. Then, as $h_0 \notin \mathfrak{P}_{(\vec{x})/k(\vec{g})}$, by specializing $X_i \mapsto x_i$ we obtain $a(\vec{Z}) \in \langle\vec{f}(\vec{x}, \vec{Z}), \vec{q}\rangle \subseteq k(\vec{x})[\vec{Z}]$ resp.

$$\pi_x(a(\vec{Z})) \in \langle\vec{f}\rangle \subseteq k(\vec{x})[\vec{Z}]/\langle\vec{q}\rangle.$$

From $a(\vec{Z}) \in k(\vec{g})[\vec{Z}]$ we conclude $a = (\iota \circ \pi_g)(a(\vec{Z})) = \pi_x(a(\vec{Z})) \in \langle\vec{f}\rangle = \mathfrak{I}$. By assumption a is contained in $(\iota \circ \pi_g)(k(\vec{g})[\vec{Z}])$, so we have $a \in \mathfrak{I} \cap \iota(k(\vec{g})[\vec{Z}]/\langle\vec{q}\rangle)$ as required.

'$\subseteq$': Let $b \in \mathfrak{I} \cap \iota(k(\vec{g})[\vec{Z}]/\langle\vec{q}\rangle)$, and fix a representation $b(\vec{X}, \vec{Z}) \in k(\vec{X})[\vec{Z}]$ of $b = (\iota \circ \pi_g)(b(\vec{x}, \vec{Z}))$. In particular, $b(\vec{X}, \vec{Z}) - b(\vec{x}, \vec{Z})$ is contained in the kernel of the specialization $X_i \mapsto x_i$, and because of $\mathrm{T}(\vec{Z}) \setminus \{1\}$ being linearly independent over $k(\vec{x})$, there is a polynomial $s(\vec{X}) \in k[\vec{X}]$ with $s(\vec{x}) \neq 0$ and

$$s(\vec{X}) \cdot (b(\vec{X}, \vec{Z}) - b(\vec{x}, \vec{Z})) \in \langle\vec{p}\rangle \cdot k(\vec{g})[\vec{X}, \vec{Z}] \subseteq \mathfrak{H}. \tag{2}$$

As b is contained in the ideal $\mathfrak{I}$, there are $a_i \in k(\vec{x})[\vec{Z}]/\langle\vec{q}\rangle$ such that $b = \sum a_i f_i$, and by passing to representatives we can conclude

$$b(\vec{x}, \vec{Z}) = \sum a_i(\vec{x}, \vec{Z}) \cdot f_i(\vec{x}, \vec{Z}) + p_0$$

for some $p_0 \in \langle\vec{q}\rangle \cdot k(\vec{x})[\vec{Z}]$ and $a_i(\vec{x}, \vec{Z}) \in k(\vec{x})[\vec{Z}]$. Writing $p_0 = \sum_{j=1}^{v} c_j(\vec{x}, \vec{Z}) q_j$ with $c_j(\vec{X}, \vec{Z}) \in k(\vec{X})[\vec{Z}]$ and choosing $t(\vec{X}) \in k[\vec{X}] \setminus \mathfrak{P}_{(\vec{x})/k}$ appropriately we obtain

$$t(\vec{X}) \cdot \left(b(\vec{X}, \vec{Z}) - \left(\sum a_i(\vec{X}, \vec{Z}) F_i + \sum c_j(\vec{X}, \vec{Z}) q_j\right)\right) \in \underbrace{\mathfrak{P}_{(\vec{x})/k(\vec{g})}}_{\subseteq k(\vec{g})[\vec{X}]} \cdot k(\vec{g})[\vec{X}, \vec{Z}].$$

From $\mathfrak{P}_{(\vec{x})/k(\vec{g})} \cdot k(\vec{g})[\vec{X}, \vec{Z}]$ being prime and $t(\vec{x}) \neq 0$ we may conclude that

$$b(\vec{X}, \vec{Z}) - \left(\sum a_i(\vec{X}, \vec{Z}) \cdot F_i + \sum c_j(\vec{X}, \vec{Z}) \cdot q_j\right) \in \mathfrak{P}_{(\vec{x})/k(\vec{g})} \cdot k(\vec{g})[\vec{X}, \vec{Z}] \subseteq \mathfrak{H}.$$

Because of $\vec{F}, \vec{q}$ being contained in $\mathfrak{H}$, now also $b(\vec{X}, \vec{Z}) \in \mathfrak{H}$ must hold.

From (2) we therefore obtain $s(\vec{X}) \cdot b(\vec{x}, \vec{Z}) \in \mathfrak{H}$, and as $\mathfrak{H}$ is saturated w. r. t. all polynomials in $k(\vec{g})[\vec{X}] \setminus \mathfrak{P}_{(\vec{x})/k(\vec{g})}$, we have $b(\vec{x}, \vec{Z}) \in \mathfrak{H}$. Since $b(\vec{x}, \vec{Z}) \in k(\vec{g})[\vec{Z}]$ this yields

$$b = (\iota \circ \pi_g)(b(\vec{x}, \vec{Z})) \in (\iota \circ \pi_g)(\mathfrak{H} \cap k(\vec{g})[\vec{Z}])$$

as required. □

From a computational point of view the characterization of $\mathfrak{I} \cap \iota(k(\vec{g})[\vec{Z}]/\langle\vec{q}\rangle)$ in Lemma 2.2 is not really satisfying, as it does not give a hint on how to determine a basis of the ideal $\mathfrak{H}$. For computing such a finite set of generators for $\mathfrak{H}$, we can make use of the following remark:

Remark: We keep the notation from Lemma 2.2. Let $\langle\vec{F},\vec{p},\vec{q}\rangle = \bigcap_{i=1}^{w} \mathfrak{Q}_i$ be an irredundant primary decomposition. Then

$$\mathfrak{H} = \bigcap_{\substack{1\leq i\leq w \\ \mathfrak{Q}_i \cap k(\vec{g})[\vec{X}] \subseteq \mathfrak{P}_{(\vec{x})/k(\vec{g})}}} \mathfrak{Q}_i.$$

Proof: According to (Eisenbud 1995, Exercise 2.3) we have

$$\mathfrak{H} = k(\vec{g})[\vec{X},\vec{Z}] \cap \left(\langle\vec{F},\vec{p},\vec{q}\rangle \cdot k(\vec{g})[\vec{X},\vec{Z}]_{k(\vec{g})[\vec{X}]\setminus\mathfrak{P}_{(\vec{x})/k(\vec{g})}}\right)$$

where as usual $k(\vec{g})[\vec{X},\vec{Z}]_{k(\vec{g})[\vec{X}]\setminus\mathfrak{P}_{(\vec{x})/k(\vec{g})}}$ denotes the localization of $k(\vec{g})[\vec{X},\vec{Z}]$ at $k(\vec{g})[\vec{X}] \setminus \mathfrak{P}_{(\vec{x})/k(\vec{g})}$. So the claim follows from (Zariski Samuel 1979, Chapter IV, Theorem 17). □

The condition $\mathfrak{Q}_i \cap k(\vec{g})[\vec{X}] \subseteq \mathfrak{P}_{(\vec{x})/k(\vec{g})}$ in the previous remark can be verified effectively my means of standard Gröbner basis techniques (cf., e. g., (Buchberger 1965, Trinks 1978, Buchberger 1985) and (Becker Weispfenning 1993, Propositions 5.38 & 6.15)). Moreover, if the required computations in $k(\vec{g})[\vec{X},\vec{Z}]$ can be performed effectively then an irredundant primary decomposition of $\langle\vec{F},\vec{p},\vec{q}\rangle$ can be computed by means of the techniques described in (Seidenberg 1974, Gianni et al. 1988, Decker et al. 1999), for instance. Finally, computing the elimination ideal $\mathfrak{H} \cap k(\vec{g})[\vec{Z}]$ in Lemma 2.2 is a standard application of Gröbner basis techniques again, and as applying $\iota \circ \pi_g$ does not provide any difficulties, in summary we have

THEOREM 2.1: *We keep the above notation. Moreover, assume that an irredundant primary decomposition of $\langle\vec{F},\vec{p},\vec{q}\rangle \subseteq k(\vec{g})[\vec{X},\vec{Z}]$ can be computed effectively. Then a finite generating set of $\mathfrak{I} \cap \iota(k(\vec{g})[\vec{Z}]/\langle\vec{q}\rangle)$ can be computed effectively.*

We want to illustrate the method just described through a simple example:

Example: Let x be transcendental over $\mathbb{Q}$, and consider the principal ideal

$$\mathfrak{I} := \langle Z^3 + Z^2 - x^3 - x^2\rangle \subseteq \mathbb{Q}(x)[Z].$$

We want to compute the restriction $\mathfrak{I} \cap \mathbb{Q}(x^2)[Z]$. For this we first have to determine a basis of the ideal $\mathfrak{P}_{(x)/\mathbb{Q}(x^2)}$: obviously the minimal polynomial $p := Z^2 - x^2$ of x over $\mathbb{Q}(x^2)$ can be used here. Denoting the given generator of $\mathfrak{I}$ by f_1, the corresponding polynomial F_1 computes to $Z^3 + Z^2 - X^3 - X^2$. As we

are dealing with a polynomial ring we have $\mathfrak{Q} = \langle 0 \rangle$, and so in our example the ideal $\langle \vec{F}, \vec{p}, \vec{q} \rangle$ is generated by

$$\left\{ Z^3 + Z^2 - X^3 - X^2, X^2 - x^2 \right\}.$$

E. g., by means of a computer algebra system like MAGMA (see (Bosma et al. 1997)) one can determine the following irredundant primary decomposition of $\langle \vec{F}, \vec{p}, \vec{q} \rangle \subseteq \mathbb{Q}(x^2)[Z]$:

$$\langle \vec{F}, \vec{p}, \vec{q} \rangle = \underbrace{\langle Z - X, X^2 - x^2 \rangle}_{=:\mathfrak{Q}_1} \cap \underbrace{\langle Z^2 + ZX + Z + X + x^2, X^2 - x^2 \rangle}_{=:\mathfrak{Q}_2}$$

By looking at the corresponding lexicographical Gröbner basis with $Z > X$ we see that $\mathfrak{Q}_i \cap \mathbb{Q}(x^2)[X] \subseteq \mathfrak{P}_{(x)/\mathbb{Q}(x^2)}$ holds for $i = 1, 2$. So in our example we have $\mathfrak{H} = \mathfrak{Q}_1 \cap \mathfrak{Q}_2$, namely $\mathfrak{H} = \langle F_1, p \rangle$. Computing the elimination ideal $\mathfrak{H} \cap \mathbb{Q}(x^2)[Z]$ with a lexicographic Gröbner basis provides no further difficulties and yields

$$\mathfrak{I} \cap \mathbb{Q}(x^2)[Z] = \langle Z^6 + 2 \cdot Z^5 + Z^4 - 2x^2 \cdot Z^3 - 2x^2 \cdot Z^2 - x^6 + x^4 \rangle.$$

3. A (counter-)example: intersecting fields

As described in (Müller-Quade Beth 1998*a*), an ideal restriction can be used to compute generators of the intersection $k(\vec{g}) \cap k(\vec{h})$ of two subfields $k(\vec{g}), k(\vec{h}) \subseteq k(\vec{x})$: it is sufficient to find a basis of the ideal

$$\underbrace{\mathfrak{P}_{(\vec{x})/k(\vec{g})}}_{\subseteq k(\vec{g})[\vec{X}]} \cap k(\vec{h})[\vec{X}] \subseteq (k(\vec{g}) \cap k(\vec{h}))[\vec{X}]. \tag{3}$$

Unfortunately, the method discussed in the previous section does not allow the computation of the intersection (3), as in general $k(\vec{h})$ is not a subfield of $k(\vec{g})$. In (Müller-Quade Beth 1998*a*) an algorithm for accomplishing this task was proposed, but a more detailed analysis shows that it actually computes the ideal $\mathfrak{P}_{(\vec{x})/k(\vec{g})} \cdot k(\vec{x})[\vec{X}] \cap k(\vec{h})[X]$ which in general does not coincide with the ideal (3):

Example: Consider the two subfields $k(\vec{g}) := \mathbb{Q}(x^3 + x^2)$ and $k(\vec{h}) := \mathbb{Q}(x^2)$ of $k(\vec{x}) := \mathbb{Q}(x)$. Then we know from the example in the previous section that

$$\mathfrak{P}_{(\vec{x})/k(\vec{g})} \cdot k(\vec{x})[\vec{X}] \cap k(\vec{h})[X] = \langle X^6 + 2 \cdot X^5 + X^4 - 2x^2 \cdot X^3 - 2x^2 \cdot X^2 - x^6 + x^4 \rangle.$$

As adjoining the coefficients of a reduced Gröbner basis of this ideal to $\mathbb{Q}$ yields the field $\mathbb{Q}(x^2)$, the algorithm from (Müller-Quade Beth 1998*a*) yields $\mathbb{Q}(x^3 + x^2) \cap \mathbb{Q}(x^2) = \mathbb{Q}(x^2)$, which is clearly wrong.

So it remains an interesting open question whether the techniques described here can be extended in such a way that they allow the computation of a system of generators of the intersection of arbitrary finitely generated extension fields.

References

Becker, T. Weispfenning, V. (1993), *Gröbner Bases: A Computational Approach to Commutative Algebra*, Vol. 141 of *Graduate Texts in Mathematics*, Springer, New York. In cooperation with Heinz Kredel.

Bosma, W., Cannon, J. Playoust, C. (1997), 'The Magma Algebra System I: The User Language', *Journal of Symbolic Computation* **24**, 235–265.

Buchberger, B. (1965), 'Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal, (An algorithm for finding a basis for the residue class ring of a zero-dimensional polynomial ideal)', Dissertation Math. Inst. Universität Innsbruck, Austria.

Buchberger, B. (1985), *Multidimensional Systems Theory*, D. Reidel, Dordrecht, chapter Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory, pp. 184–232.

Decker, W., Greuel, G.-M. Pfister, G. (1999), Primary Decomposition: Algorithms and Comparisons, *in* B. H. Matzat, G.-M. Greuel G. Hiss, eds, 'Algorithmic Algebra and Number Theory', Springer, pp. 187–220.

Eisenbud, D. (1995), *Commutative Algebra with a View Toward Algebraic Geometry*, Vol. 150 of *Graduate Texts in Mathematics*, Springer, New York.

Gianni, P., Trager, B. Zacharias, G. (1988), 'Gröbner Bases and Primary Decomposition of Polynomial Ideals', *Journal of Symbolic Computation* **6**, 149–167.

Kapur, D. Madlener, K. (1989), A Completion Procedure for Computing a Canonical Basis for a k-subalgebra, *in* 'Computers and Mathematics', Springer, pp. 1–11.

Kemper, G. (1993), 'An Algorithm to Determine Properties of Field Extensions Lying over a Ground Field', IWR Preprint 93-58, Heidelberg.

Müller-Quade, J. Beth, T. (1998*a*), 'Computing the Intersection of Finitely Generated Fields', Poster presented at the International Symposium on Symbolic and Algebraic Computation, ISSAC'98. Abstract available in (Müller-Quade Beth 1998*b*).

Müller-Quade, J. Beth, T. (1998*b*), 'Computing the Intersection of Finitely Generated Fields', *SIGSAM Bulletin* **32**(2), 62–62. Abstract.

Müller-Quade, J. Rötteler, M. (1998), Deciding Linear Disjointness of Finitely Generated Fields, *in* O. Gloor, ed., 'Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation', The Association for Computing Machinery, Inc. (ACM), pp. 153–160.

Müller-Quade, J. Steinwandt, R. (1999), 'Basic Algorithms for Rational Function Fields', *Journal of Symbolic Computation* **27**(2), 143–170.

Müller-Quade, J. Steinwandt, R. (2000), 'Gröbner bases applied to finitely generated field extensions', *Journal of Symbolic Computation* **30**(4), 469–490.

Müller-Quade, J., Steinwandt, R. Beth, T. (1998), An application of Gröbner bases to the decomposition of rational mappings, *in* B. Buchberger F. Winkler, eds, 'Gröbner Bases and Applications', Vol. 251 of *Lecture Note Series*, London Mathematical Society, Cambridge University Press, pp. 448–462.

Robbiano, L. Sweedler, M. (1990), Subalgebra bases, *in* W. Bruns A. Simis, eds, 'Commutative Algebra', Vol. 1430 of *Lecture Notes in Mathematics*, Springer, pp. 61–87.

Robbiano, L. Sweedler, M. (1998), 'Ideal and Subalgebra Coefficients', *Proc. Am. Math. Soc.* **126**(8), 2213–2219.

Seidenberg, A. (1974), 'Constructions in Algebra', *Trans. Am. Math. Soc.* **197**, 273–313.

Steinwandt, R. Müller-Quade, J. (2000), On restricting ideals in finitely generated k-algebras, *in* T. Mulders, ed., 'Proceedings of the Seventh Rhine Workshop on Computer Algebra RWCA'00', pp. 119–124.

Sweedler, M. (1993), Using Groebner Bases to Determine the Algebraic and Transcendental Nature of Field Extensions: return of the killer tag variables, *in* G. Cohen, T. Mora O. Moreno, eds, 'Applied Algebra, Algebraic Algorithms and Error-Correcting Codes 10th International Symposium, AAECC-10', Vol. 673 of *Lecture Notes in Computer Science*, Springer, Berlin; Heidelberg, pp. 66–75.

Trinks, W. (1978), 'Über B. Buchbergers Verfahren, Systeme algebraischer Gleichungen zu lösen', *Journal of Number Theory* **10**, 475–488.

Zariski, O. Samuel, P. (1979), *Commutative Algebra — Volume I*, Graduate Texts in Mathematics, second edn, Springer, New York; Heidelberg; Berlin.

New rewriting system for the braid group $\mathcal{B}_4$

Leonid Bokut[1] and Andrei Vesnin[1,2]

[1] *Sobolev Institute of Mathematics, Novosibirsk 630090, Russia*

[2] *School of Mathematical Sciences, Seoul National University, Seoul 151-747, Korea*

To 60-th birthday of a pioneer of Gröbner bases Prof. B. Buchberger

Abstract

Using presentations of the braid groups $\mathcal{B}_3$ and $\mathcal{B}_4$ as towers of HNN extensions of the free group of rank 2, we obtain normal forms, Gröbner bases and rewriting systems for these groups.

KEYWORDS: braid group, normal form, Gröbner basis

1. Introduction

In this note we obtain Gröbner bases as well as rewriting systems for braid groups $\mathcal{B}_3$ and $\mathcal{B}_4$. Braid groups are subject of the intensive studding in group theory and low-dimensional topology. We refer to (Birman 1974) for the basic properties of braid groups. It was shown by P. Dehornoy that braid groups are left-orderable (Dehornoy 2000), and different kinds of normal forms and rewriting systems for braid groups can be found in (Birman et al. 1998, Elrifai Morton 1994, Garside 1969, Garber et al. 2002, Hermiller Meier 1999, Markov 1945, Thurston 1992).

Our interest in Gröbner bases for braid groups is motivated by the close relation between non-commutative Gröbner bases and rewriting systems for semigroups. This relation is a kind of a folklore, and was fully described, for example, in (Heyworth 2000). As an introduction to string rewriting we refer to (Book Otto 1993). The basis facts about Gröbner bases will be presented in Section 2.

One of aims of this note is to develop and to demonstrate a method of constructing of Gröbner bases, normal forms, and rewriting systems for groups, presented as towers of HNN extensions of free groups. We realize this method for braid groups $\mathcal{B}_3$ and $\mathcal{B}_4$.

*Authors were supported in part by the Russian Foundation for Basic Research (grants 01–01–00630 and 02–01–01118).

In Section 3, applying the Magnus-Moldovanski rewriting procedure (see Lyndon Schupp (1977)) for $\mathcal{B}_3$ and $\mathcal{B}_4$ (which is 3-relator group rather than to 1-relator classical case) we obtain presentations of these groups as towers of HNN extensions of the free group of rank 2. Obtained presentations are closely related to the presentations of commutators $\mathcal{B}_3'$ and $\mathcal{B}_4'$ which have been find by E. Gorin and V. Lin (see Gorin Lin (1969)) using the Reidemeister-Shreier method. So, we will refer to the generators as *Gorin-Lin generators*.

These presentations lead to the standard normal forms in $\mathcal{B}_3$ and $\mathcal{B}_4$, and to the standard rewriting systems for them in the sense of (Bokut 1966, 1967, 1980). We show it in Section 4, using the Gröbner–Shirshov bases technic unlike of a group-theoretic technic of (Bokut 1966, 1967). More precisely, we give the Gröbner–Shirshov basis of $\mathcal{B}_4$ (and as a corollary, for $\mathcal{B}_3$) in Gorin-Lin generators and relative to an appropriate order of group words, the tower order. This order was implicitly used in (Bokut 1966, 1967). The applications of similar technique to Coxeter groups and Novikov and Boon groups can be found in (Bokut Shiao 2001, 2002).

From the normal form, it readily follows some known properties of $\mathcal{B}_4$ proved in the above mentioned paper by E. Gorin and V. Lin, in particular, the presentation of the commutator group $\mathcal{B}_4'$ and its description as the semidirect product of two free groups of rank two. Moreover, the relation of obtained presentations of braid groups with cyclically presented groups, such as Sieradski groups and Fibonacci groups is discussed.

This work was done during the first author visit to the KIAS, Seoul. We thank Dr. Sang-Jin Lee for very helpful discussions.

2. Non-Commutative Gröbner bases

In this section we recall some facts about *non-commutative Gröbner bases* which are also known in the literature as *Gröbner–Shirshov bases* (see, for example, Ufnarovski (1998)).

Let X be a linearly ordered set, k a field, $k\langle X\rangle$ the free associative algebra over X and k. On the set X^* of words we impose a well order "$>$" that is compatible with the concatenations of words. For example, it may be the deglex order (compare two words first by degrees and then lexicographically) or the tower order below. Any polynomial $f \in k\langle X\rangle$ has the *leading word* $\bar{f}$ relative to "$>$".

We say that f is *monic* if $\bar{f}$ occurs in f with coefficient 1. By a *composition of intersection* $(f,g)_w$ of two monic polynomials relative to some word w, such that $w = \bar{f}b = a\bar{g}$, $\deg(\bar{f}) + \deg(\bar{g}) > \deg(w)$, one means the following polynomial

$$(f,g)_w = fb - ag.$$

By *composition of including* $(f,g)_w$ of two monic polynomials, where $w = \bar{f} = a\bar{g}b$, one means the following polynomial

$$(f,g)_w = f - agb.$$

In the last case the transformation

$$f \to (f,g)_w = f - agb$$

is called the *elimination of the leading word* (ELW) of g in f.

A composition $(f,g)_w$ is called *trivial* relative to some $R \subset k\langle X\rangle$ and w (we write it as $(f,g)_w \equiv 0\, mod(R,w)$) if

$$(f,g)_w = \sum \alpha_i a_i t_i b_i,$$

where $\alpha_i \in k$, $t_i \in R$, $a_i, b_i \in X^*$, and $a_i\bar{t}_i b_i < w$. In particular, if $(f,g)_w$ goes to zero by the ELW's of R then $(f,g)_w$ is trivial relative to R.

For two polynomials f_1 and f_2 we write

$$f_1 \equiv f_2\, mod(R,w)$$

if and only if

$$f_1 - f_2 \equiv 0\, mod(R,w).$$

A subset R of $k\langle X\rangle$ is called *Gröbner–Shirshov basis* if any composition of polynomials from R is trivial relative to R.

By $\langle X|R\rangle$, the algebra with generators X and defining relations R, we will mean the factor-algebra of $k\langle X\rangle$ by the ideal generated by R.

The following lemma goes back to the Poincare-Birkgoff-Witt theorem, the Diamond Lemma of M.H.A. Newman (Newman 1942), the Composition Lemma of A.I. Shirshov (Shirshov 1962) (see also Bokut (1972, 1976), where this Composition Lemma was formulated explicitly and in a current form), the Buchberger's Theorem (Buchberger (1965), published in Buchberger (1970)), the Diamond Lemma of G. Bergman (Bergman 1978) (this lemma was also known to P.M. Cohn (see, for example, Cohn (1966)) and some historical comments to Chapter "Gröbner basis" in (Eisenbud 1995)):

Composition–Diamond Lemma. R is a Gröbner–Shirshov basis if and only if the set

$$PBW(R) = \{u \in X^* \mid u \neq a\bar{f}b, \text{ for any } f \in R\}$$

of R-reduced words consists of a linear basis of the algebra $\langle X|R\rangle$.

The set $PBW(R)$ will be called the *PBW-basis* of $\langle X|R\rangle$ relative to a Gröbner–Shirshov basis R.

If a subset R of $k\langle X\rangle$ is not a Gröbner–Shirshov basis then one can add to R all non trivial compositions of polynomials of R, and continue this process (infinitely) many times in order to have a Gröbner–Shirshov basis R^{comp} that contains R.

A Gröbner–Shirshov basis R is called *reduced* if any $s \in R$ is a linear combination of $R \setminus \{s\}$–reduced words. Any ideal of $k\langle X\rangle$ has a unique reduced Gröbner–Shirshov basis.

If R is a set of "semigroup relations" (that is, polynomials of the form $u - v$, where $u, v \in X^*$), then any non trivial composition will have the same form. As the result the set R^{comp} consists of semigroup relations too.

Let $A = smg\langle X|R\rangle$ be a semigroup presentation. Then R is a subset of $k\langle X\rangle$ and one can find a Gröbner–Shirshov basis R^{comp}. The last set does not depend of k, and consists of semigroup relations. We will call R^{comp} to be a Gröbner–Shirshov basis of A. It is the same as a Gröbner–Shirshov basis of the semigroup algebra $kA = \langle X|R\rangle$.

The same terminology is valid for any group presentation meaning that we include in this presentation all trivial group relations of the form

$$xx^{-1} = 1, x^{-1}x = 1, x \in X.$$

3. Presentations of groups $\mathcal{B}_3$ and $\mathcal{B}_4$

In this section we will obtain presentations of braid groups as towers of HNN extensions of the free group of rank two.

We start from the consideration of the 3-strand braid group $\mathcal{B}_3$ with the following presentation:

$$\mathcal{B}_3 = \langle \sigma_1, \sigma_2 \mid \sigma_1\sigma_2\sigma_1 = \sigma_2\sigma_1\sigma_2 \rangle. \tag{1}$$

Let us denote $t = \sigma_1$ and consider y_2 such that $\sigma_2 = y_2 t$, i.e. $y_2 = \sigma_2\sigma_1^{-1}$. Then we have

$$\mathcal{B}_3 \cong \langle y_2, t \mid y_2 t t y_2 = t y_2 t \rangle.$$

Introducing notation $y_{2(i)} = t^i y_2 t^{-i}$ for $i = 1, 2$, we get

$$\mathcal{B}_3 \cong \langle y_2, y_{2(1)}, y_{2(2)}, t \mid y_2 y_{2(2)} = y_{2(1)}, \quad y_{2(1)}t = ty_2, \quad y_{2(2)}t = ty_{2(1)} \rangle.$$

Eliminating $y_{2(2)}$ using the first relation, we will obtain:

$$\mathcal{B}_3 \cong \langle y_2, y_{2(1)}, t \mid y_{2(1)}t = ty_2, \quad y_2^{-1}y_{2(1)}t = ty_{2(1)} \rangle$$

and finally,

$$\mathcal{B}_3 \cong \langle y_2, y_{2(1)}, t \mid y_{2(1)}t = ty_2, \quad y_2 t = t y_2 y_{2(1)}^{-1} \rangle,$$

where we used the first relation to modify the second.

Let us rewrite this relation using notations $y_2 = t_2$ and $y_{2(1)} = t_1$:

$$\mathcal{B}_3 \cong \langle t_1, t_2, t \mid t_1 t = t t_2, \quad t_2 t = t t_2 t_1^{-1} \rangle. \tag{2}$$

Thus, we have the following extension:

$$\langle t_1, t_2 \rangle \subset \mathcal{B}_3 = \langle t_1, t_2, t \rangle, \tag{3}$$

that can be regarded as an HNN-extension (by the conjugation automorphism t) of the free group with two generators. The relation of these generators with the standard generators of $\mathcal{B}_3$ is the following: $t_2 = \sigma_2\sigma_1^{-1}$ and $t_1 = \sigma_1\sigma_2\sigma_1^{-2}$. Braids corresponding to t, t_1, and t_2 are presented in the following figure.

t :  t_1 :  t_2 : 

As the result, the kernel of the homomorphism from $\mathcal{B}_3$ to $\langle t\rangle$ defined by $t \mapsto t$, $t_1 \mapsto 1$, $t_2 \mapsto 1$, is the free group $\langle t_1, t_2\rangle$. This kernel coincides with the commutant $\mathcal{B}_3'$, that was also shown in (Gorin Lin 1969) by using of the Reidemeister–Schreier method. So, we will refer to t, t_1 and t_2 as *Gorin-Lin generators* of $\mathcal{B}_3$.

From the defining relations (2) and their inverses we obtain relations:

$$\begin{array}{ll} t_2t = tt_2t_1^{-1}, & t_1t^{-1} = t^{-1}t_2^{-1}t_1, \\ t_1t = tt_2, & t_2t^{-1} = t^{-1}t_1, \end{array}$$

where we used

$$t_2t = tt_2t_1^{-1} \Leftrightarrow t_2t = t_1tt_1^{-1} \Leftrightarrow t_1^{-1}t_2t = tt_1^{-1}, \Leftrightarrow t_1t^{-1} = t^{-1}t_2^{-1}t_1.$$

In the next section we will prove that the following transformations give rise the rewriting system for $\mathcal{B}_3$ (that will follows from the rewriting system for $\mathcal{B}_4$ and embedding $\mathcal{B}_3 \subset \mathcal{B}_4$):

$$\begin{array}{ll} (t_2)^{\pm1}t \longrightarrow t(t_2t_1^{-1})^{\pm1}, & (t_1)^{\pm1}t^{-1} \longrightarrow t^{-1}(t_2^{-1}t_1)^{\pm1}, \\ (t_1)^{\pm1}t \longrightarrow t(t_2)^{\pm1}, & (t_2)^{\pm1}t^{-1} \longrightarrow t^{-1}(t_1)^{\pm1}. \end{array}$$

It gives the following normal form for $\mathcal{B}_3$:

$$t^nV(t_1, t_2)$$

where $n \in \mathbb{Z}$ and $V(t_1, t_2)$ denotes a group word in t_1, t_2.

We would like to point out the following relation of the braid group presentation (2) with cyclically presented groups (in sense of Johnson (1980)).

Consider the conjugation action of t on the group $\mathcal{B}_3' = \langle t_1, t_2\rangle$. Let us denote $a_0 = t_2$ and $a_i = t^ia_0t^{-i}$ for $i \in \mathbb{Z}$. In particular, we get $a_1 = t_1$ and $a_{-1} = t_2t_1^{-1} = a_0a_1^{-1}$. Therefore, for each i we have $a_ia_{i+2} = a_{i+1}$, and the following group presentation with infinite number of generators naturally arises:

$$G_\infty = \langle a_i,\, i \in \mathbb{Z} \mid a_ia_{i+2} = a_{i+1}, \quad i \in \mathbb{Z}\rangle$$

For a reader convenience we recall the expression of a_i in terms of the Artin generators (1):

$$a_i = \sigma_1^i\sigma_2\sigma_1^{-(i+1)}.$$

Remark that the "truncated" version of this group, i.e.

$$G_n = \langle a_1, \ldots, a_n \mid a_ia_{i+2} = a_{i+1}, \quad i = 1, \ldots, n\rangle,$$

where all suffices are taken by mod n, is known as the *Sieradski group* (see, for example, Cavicchioli et al. (1998)) and is the fundamental group of the n-fold cyclic branched covering of the 3-sphere, branched over the trefoil knot.

Now let us consider the 4-strand braid group $\mathcal{B}_4$ with the following presentation:

$$\mathcal{B}_4 = \langle \sigma_1, \sigma_2, \sigma_3 \mid \sigma_2\sigma_1\sigma_2 = \sigma_1\sigma_2\sigma_1, \quad \sigma_3\sigma_2\sigma_3 = \sigma_2\sigma_3\sigma_2, \quad \sigma_3\sigma_1 = \sigma_1\sigma_3 \rangle. \quad (4)$$

Let us denote $t = \sigma_1$. Consider y_3 such that $\sigma_3 = y_3 t$, i.e. $y_3 = \sigma_3\sigma_1^{-1}$, and y_2 such that $\sigma_2 = y_2 t$, i.e. $y_2 = \sigma_2\sigma_1^{-1}$. Then we have

$$\mathcal{B}_4 \cong \langle y_2, y_3, t \mid y_2 t t y_2 = t y_2 t, \quad y_3 t y_2 t y_3 = y_2 t y_3 t y_2, \quad y_3 t = t y_3 \rangle.$$

As well as above, let us introduce $y_{j(i)} = t^i y_j t^{-i}$ for $j = 2, 3$ and $i \in \mathbb{Z}$. Then

$$\begin{aligned}\mathcal{B}_4 \cong \langle y_2, y_{2(1)}, y_{2(2)}, y_3, t \mid \quad & y_2 y_{2(2)} = y_{2(1)}, \quad y_3 y_{2(1)} y_3 = y_2 y_3 y_{2(2)}, \\ & y_3 t = t y_3, \quad y_{2(1)} t = t y_2, \quad y_{2(2)} t = t y_{2(1)} \rangle,\end{aligned}$$

where we used that t and y_3 commute. Eliminating $y_{2(2)}$ using the first defining relation, we will obtain:

$$\begin{aligned}\mathcal{B}_4 \cong \langle y_2, y_{2(1)}, y_3, t \quad \mid \quad & y_3 y_{2(1)} y_3 = y_2 y_3 y_2^{-1} y_{2(1)}, \quad y_3 t = t y_3, \\ & y_{2(1)} t = t y_2, \quad y_2^{-1} y_{2(1)} t = t y_{2(1)} \rangle.\end{aligned}$$

Denoting $t_1 = y_{2(1)}$, $t_2 = y_2$ and $b = y_3$ we get

$$\mathcal{B}_4 \cong \langle t_1, t_2, t, b \quad \mid \quad b t_1 b = t_2 b t_2^{-1} t_1,\ b t = t b,\ t_1 t = t t_2,\ t_2^{-1} t_1 t = t t_1 \rangle,$$

and so,

$$\mathcal{B}_4 \cong \langle t_1, t_2, t, b \quad \mid \quad t_1 b = b^{-1} t_2 b t_2^{-1} t_1,\ b t = t b,\ t_1 t = t t_2,\ t_2 t = t t_2 t_1^{-1} \rangle,$$

where we used the third relation to modify the forth relation. Denoting $a = t_1 b t_1^{-1}$, we get

$$\begin{aligned}\mathcal{B}_4 \cong \langle a, b, t_1, t_2, t \quad \mid \quad & t_1 t = t t_2, \quad t_2 t = t t_2 t_1^{-1}, \quad b t = t b, \\ & a t_1 = t_1 b, \quad a = b^{-1} t_2 b t_2^{-1} \rangle.\end{aligned} \quad (5)$$

Generators t, t_1, t_2, a, b will be referred to as *Gorin-Lin generators* of $\mathcal{B}_4$.

Using above defining relations we have

$$a t = t_1 b t_1^{-1} t = t_1 b t t_2^{-1} = t_1 t b t_2^{-1} = t t_2 b t_2^{-1} = t b a. \quad (6)$$

Let us multiply both sides of the relation

$$t_1 b = b^{-1} t_2 b t_2^{-1} t_1$$

from the left by t, and remark that after that the left part can be modified as

$$tt_1b = t_2^{-1}tt_2b = t_2^{-1}t_1tb = t_2^{-1}t_1bt = t_2^{-1}at_1t,$$

and the right part can be modified as

$$tb^{-1}t_2bt_2^{-1}t_1 = b^{-1}tt_2bt_2^{-1}t_1 = b^{-1}t_1tbt_2^{-1}t_1 = b^{-1}t_1btt_2^{-1}t_1$$
$$= b^{-1}t_1bt_1^{-1}tt_1 = b^{-1}at_2^{-1}tt_2 = b^{-1}at_2^{-1}t_1t.$$

Thus,

$$t_2^{-1}a = b^{-1}at_2^{-1},$$

that gives us

$$at_2 = t_2b^{-1}a. \tag{7}$$

Using this result we have

$$a = b^{-1}t_2bt_2^{-1} \iff bat_2 = t_2b \iff bt_2b^{-1}a = t_2b \iff bt_2 = t_2ba^{-1}b. \tag{8}$$

Now let us conjugate both sides of the obtained relation by t. Then from the left part of the relation we will get

$$tbt_2t^{-1} = btt_2t^{-1} = bt_1,$$

and from the right part of the relation we will get

$$tt_2ba^{-1}bt^{-1} = t_1tba^{-1}t^{-1}b = t_1bta^{-1}t^{-1}b = t_1ba^{-1}tbt^{-1}b = t_1ba^{-1}b^2.$$

Hence

$$bt_1 = t_1ba^{-1}b^2. \tag{9}$$

Summarizing (5), (6), (7), (8) and (9) we get

LEMMA 3.1: *The following relations are valid for Gorin-Lin generators of $\mathcal{B}_4$:*

$$t_2t = tt_2t_1^{-1} \qquad t_1t = tt_2, \qquad bt = tb, \qquad at = tba,$$
$$at_1 = t_1b, \qquad bt_1 = t_1ba^{-1}b^2, \qquad at_2 = t_2b^{-1}a, \qquad bt_2 = t_2ba^{-1}b,$$

and for their inverses:

$$t_1t^{-1} = t^{-1}t_2^{-1}t_1, \quad t_2t^{-1} = t^{-1}t_1, \quad bt^{-1} = t^{-1}b, \quad at^{-1} = t^{-1}b^{-1}a,$$
$$bt_1^{-1} = t_1^{-1}a, \quad at_1^{-1} = t_1^{-1}a^2b^{-1}a, \quad bt_2^{-1} = t_2^{-1}ba, \quad at_2^{-1} = t_2^{-1}ba^2.$$

Therefore, we have the description of $\mathcal{B}_4$ as a tower of HNN extensions of the free group of rank two:

$$\langle a, b\rangle \subset \langle a, b, t_1, t_2\rangle \subset \mathcal{B}_4 = \langle a, b, t_1, t_2, t\rangle.$$

The correspondence between Gorin-Lin generators and the standard generators of $\mathcal{B}_4$ is the following:

$$t = \sigma_1, \quad t_1 = \sigma_1\sigma_2\sigma_1^{-2}, \quad t_2 = \sigma_2\sigma_1^{-1}, \quad a = \sigma_1\sigma_2\sigma_1^{-1}\sigma_3\sigma_2^{-1}\sigma_1^{-1}, \quad b = \sigma_3\sigma_1^{-1}.$$

Remark that t, t_1 and t_2 generate such a subgroup of $\mathcal{B}_4$ with the presentation (2) that gives us $\mathcal{B}_3$. Braids corresponding to t, t_1, t_2, a and b are presented in the following figure.

In the next section we will find a rewriting system for $\mathcal{B}_4$. As the result, a normal form for $\mathcal{B}_4$ is given by expression

$$t^n V(t_1, t_2) W(a, b),$$

where $n \in \mathbb{Z}$, and $V(t_1, t_2)$, $W(a, b)$ are free group words in alphabets $\{t_1, t_2\}$ and $\{a, b\}$, respectively. The commutant $\mathcal{B}_4'$, the kernel of the homomorphism from $\mathcal{B}_4$ to $\langle t \rangle$, defined by

$$t \mapsto t, \quad t_1 \mapsto 1, \quad t_2 \mapsto 1, \quad a \mapsto 1, \quad b \mapsto 1$$

is the semi-direct product of free groups $\langle t_1, t_2 \rangle$ and $\langle a, b \rangle$. Here $\langle t_1, t_2 \rangle$ is the commutant of $\mathcal{B}_3$. The kernel of the homomorphism of $\mathcal{B}_4$ to $\mathcal{B}_3$, defined by

$$\sigma_1 \mapsto \sigma_1, \qquad \sigma_2 \mapsto \sigma_2, \qquad \sigma_3 \mapsto \sigma_1,$$

that is by

$$t \mapsto t, \qquad t_1 \mapsto t_1, \qquad t_2 \mapsto t_2, \qquad a \mapsto 1, \qquad b \mapsto 1,$$

is the free group $\langle a, b \rangle$. Remark that in (Gorin Lin 1969) all these facts were proved using the Reidemeister-Shreier method, where $t^{-1}at$ and b were chosen as the generators of this free group.

Now let us point out the connection of relations from Lemma 3.1 with Fibonacci groups $F(2, n)$ studied by many authors.

Consider the conjugation action of t_1 on $\langle a, b \rangle$:

$$t_1^{-1} a t_1 = b, \qquad t_1^{-1} b t_1 = b a^{-1} b^2.$$

Let us denote $z_0 = b$ and $z_i = t_1^{-i} z_0 t_1^i$ for $i \in \mathbb{Z}$. Thus, we get $a = z_{-1}$ and $z_1 = z_0 z_{-1}^{-1} z_0^2$. Therefore for $i \in \mathbb{Z}$ we have

$$z_{i+1} z_i^{-1} z_{i+1}^2 = z_{i+2},$$

and the following group with infinite number of generators naturally arise:

$$H_\infty = \langle z_i,\, i \in \mathbb{Z} \,|\, z_{i+1} z_i^{-1} z_{i+1}^2 = z_{i+2}, \quad i \in \mathbb{Z} \rangle.$$

Remark that this group presentation coincides with the presentation of the commutator subgroup of the figure-eight knot group obtained in (Burde Zieschang 1985, p. 35) and action of t_1 on a and b corresponds to the presentation of the figure-eight knot as a fibred knot (Burde Zieschang 1985, p. 73).

Similar to (Kim et al. 2000), rewrite the defining relations as

$$(z_i^{-1}z_{i+1})\, z_{i+1} = (z_{i+1}^{-1}z_{i+2}), \quad i \in \mathbb{Z}$$

and denote $x_{2i-1} = z_i$ and $x_{2i} = z_i^{-1}z_{i+1}$ for $i \in \mathbb{Z}$. Then the above relations can be rewritten as $x_{2i} = x_{2i-1}^{-1}x_{2i+1}$ and $x_{2i}x_{2i+1} = x_{2i+2}$. Then

$$H_\infty \cong \langle x_{2i-1}, x_{2i}, i \in \mathbb{Z} \,|\, x_i x_{i+1} = x_{i+2},\, i \in \mathbb{Z}\rangle.$$

For a reader convenience we recall the expression of x_i in terms of the Gorin-Lin generators:

$$x_{2i-1} = t_1^{-i}bt_1^i, \qquad x_{2i} = t_1^{-i}b^{-1}t_1^{-1}bt_1t_1^i, \qquad i \in \mathbb{Z}.$$

Thus, relations $x_{2i-1}x_{2i} = x_{2i+1}$ follow immediately, and relations $x_{2i}x_{2i+1} = x_{2i+2}$ follow from the relation $bt_1 = t_1bt_1b^{-1}t_1^{-1}b^2$.

The "truncated" version of this group is well-known as the *Fibonacci group*:

$$F(2,2n) = \langle x_1, \ldots, x_{2n} \,|\, x_i x_{i+1} = x_{i+2}, \quad i = 1, \ldots, 2n\rangle$$

where all indices are taken by mod $2n$. For any $n \geq 2$ the group $F(2,2n)$ is the fundamental group of a 3-dimensional manifold (see Helling et al. (1998)) which can be described as the n-fold cyclic branched cover of the 3-sphere branched over the figure-eight knot. From the above considerations one can easy obtain expressions of generators of Fibonacci groups in terms of Artin generators of $\mathcal{B}_4$.

4. Rewriting systems for $\mathcal{B}_3$ and $\mathcal{B}_4$

With each of the relations of $\mathcal{B}_4$ from Lemma 3.1 we associate another relation in the natural way shown below. Thus, we will get pairs of relations as follows:

$$t_2t = tt_2t_1^{-1} \quad \text{and} \quad t_2^{-1}t = tt_1t_2^{-1} \tag{10}$$
$$t_1t = tt_2 \quad \text{and} \quad t_1^{-1}t = tt_2^{-1} \tag{11}$$
$$bt = tb \quad \text{and} \quad b^{-1}t = tb^{-1} \tag{12}$$
$$at = tba \quad \text{and} \quad a^{-1}t = ta^{-1}b^{-1} \tag{13}$$
$$at_1 = t_1b \quad \text{and} \quad a^{-1}t_1 = t_1b^{-1} \tag{14}$$
$$bt_1 = t_1ba^{-1}b^2 \quad \text{and} \quad b^{-1}t_1 = t_1b^{-2}ab^{-1} \tag{15}$$
$$at_2 = t_2b^{-1}a \quad \text{and} \quad a^{-1}t_2 = t_2a^{-1}b \tag{16}$$
$$bt_2 = t_2ba^{-1}b \quad \text{and} \quad b^{-1}t_2 = t_2b^{-1}ab^{-1} \tag{17}$$
$$t_1t^{-1} = t^{-1}t_2^{-1}t_1 \quad \text{and} \quad t_1^{-1}t^{-1} = t^{-1}t_1^{-1}t_2 \tag{18}$$
$$t_2t^{-1} = t^{-1}t_1 \quad \text{and} \quad t_2^{-1}t^{-1} = t^{-1}t_1^{-1} \tag{19}$$
$$bt^{-1} = t^{-1}b \quad \text{and} \quad b^{-1}t^{-1} = t^{-1}b^{-1} \tag{20}$$
$$at^{-1} = t^{-1}b^{-1}a \quad \text{and} \quad a^{-1}t^{-1} = t^{-1}a^{-1}b \tag{21}$$
$$bt_1^{-1} = t_1^{-1}a \quad \text{and} \quad b^{-1}t_1^{-1} = t_1^{-1}a^{-1} \tag{22}$$
$$at_1^{-1} = t_1^{-1}a^2b^{-1}a \quad \text{and} \quad a^{-1}t_1^{-1} = t_1^{-1}a^{-1}ba^{-2} \tag{23}$$
$$bt_2^{-1} = t_2^{-1}ba \quad \text{and} \quad b^{-1}t_2^{-1} = t_2^{-1}a^{-1}b^{-1} \tag{24}$$
$$at_2^{-1} = t_2^{-1}ba^2 \quad \text{and} \quad a^{-1}t_2^{-1} = t_2^{-1}a^{-2}b^{-1} \tag{25}$$

Let S be the set of above listed relations (10) – (25) together with trivial relations $xx^{-1}=1$, and $x^{-1}x=1$ for $x\in\{a,b,t_1,t_2,t\}$.

We will prove that S is the Gröbner–Shirshov basis of $\mathcal{B}_4$ relative to the following tower order of words. Let us order group words in a,b by the deg-lex order. Any group word in a, b, t_2 has a form

$$u=u_0t_2^{\varepsilon_1}\cdots u_kt_2^{\varepsilon_k}u_{k+1},$$

where $u_i\in\langle a,b\rangle$, $k\geq 0$, $\varepsilon_i=\pm 1$. Define

$$wt(u)=(k,u_0,t_2^{\varepsilon_1},\ldots,u_k,t_2^{\varepsilon_k},u_{k+1}).$$

Let us order wt's lexicographically assuming $t^{-1}<t$. Let us define the tower order

$$u>_{tow}v \quad \text{if and only if} \quad wt(u)>_{lex}wt(v).$$

In the same way we can define the tower order for group words with the extra letter t_1 and then for group words with the extra letter t.

THEOREM 4.1: *The above described set S is the reduced Gröbner–Shirshov basis for $\mathcal{B}_4$ in the Gorin-Lin generators relative to the tower order of group words.*

Proof: To proof the statement we need to check that all compositions (in the sense of Section 2) of pairs of elements of S are trivial. Here we will do it for few of them. For all others similar considerations are used, but we omit them here.

Let us consider the composition $(10_\ell)\wedge(16_\ell)$ of an intersection of left relations in (10) and (16) relative to a word $w=at_2t$. We have $g=t_2t-tt_2t_1^{-1}$ with the leading word $\bar{g}=t_2t$ and $f=at_2-t_2b^{-1}a$ with the leading word $\bar{f}=at_2$. Therefore $w=a\bar{g}=\bar{f}t$. Then

$$\begin{aligned}(10_\ell)\wedge(16_\ell) &= (f,g)_w=ft-ag=(at_2-t_2b^{-1}a)t-a(t_2t-tt_2t_1^{-1})\\ &= att_2t_1^{-1}-t_2b^{-1}at\equiv tbat_2t_1^{-1}-t_2b^{-1}tba\\ &\equiv tbt_2b^{-1}at_1^{-1}-t_2tb^{-1}ba\equiv tt_2ba^{-1}bb^{-1}at_1^{-1}-t_2ta\\ &\equiv tt_2bt_1^{-1}-tt_2t_1^{-1}a\equiv tt_2t_1^{-1}a-tt_2t_1^{-1}a\equiv 0,\end{aligned}$$

where we used relations from the set S for the ELW's of S.

Now, let us check that the composition $(10_r)\wedge(24_\ell)$ of right relation of (10) and left relation of (24) is trivial with respect to S and $w=bt_2^{-1}t$. We have $g=t_2^{-1}t-tt_1t_2^{-1}$ with the leading word $\bar{g}=t_2^{-1}t$ and $f=bt_2^{-1}-t_2^{-1}ba$ with the leading word $\bar{f}=bt_2^{-1}$. Therefore $w=b\bar{g}=\bar{f}t$. Then

$$\begin{aligned}(10_r)\wedge(24_\ell) &= (f,g)_w=ft-bg=(bt_2^{-1}-t_2^{-1}ba)t-b(t_2^{-1}t-tt_1t_2^{-1})\\ &= btt_1t_2^{-1}-t_2^{-1}bat\equiv tbt_1t_2^{-1}-t_2^{-1}btba\\ &\equiv tt_1ba^{-1}b^2t_2^{-1}-t_2^{-1}tb^2a\equiv tt_1ba^{-1}bt_2^{-1}ba-tt_1t_2^{-1}b^2a\\ &\equiv tt_1ba^{-1}t_2^{-1}baba-tt_1t_2^{-1}b^2a\equiv tt_1bt_2^{-1}a^{-2}b^{-1}baba-tt_1t_2^{-1}b^2a\\ &\equiv tt_1t_2^{-1}baa^{-1}ba-tt_1t_2^{-1}b^2a\equiv tt_1t_2^{-1}b^2a-tt_1t_2^{-1}b^2a\equiv 0,\end{aligned}$$

where we used relations from the set S for the rewriting process. By the similar considerations for other compositions, the statement of the theorem holds.

It easy to see, that any $s \in S$ is a difference of $S \setminus \{s\}$ –reduced words. It means that S is a reduced Gröbner–Shirshov basis. □

The above Gröbner–Shirshov basis of $\mathcal{B}_4$ gives rise to the rewriting system (semi-Thue system) for $\mathcal{B}_4$ that is defined by rules:

$$
\begin{array}{ll}
t_2^{\pm 1}t \longrightarrow t(t_2t_1^{-1})^{\pm 1}, & t_1^{\pm 1}t \longrightarrow tt_2^{\pm 1}, \\
b^{\pm 1}t \longrightarrow tb^{\pm 1}, & a^{\pm 1}t \longrightarrow t(ba)^{\pm 1}, \\
a^{\pm 1}t_1 \longrightarrow t_1b^{\pm 1}, & b^{\pm 1}t_1 \longrightarrow t_1(ba^{-1}b^2)^{\pm 1}, \\
a^{\pm 1}t_2 \longrightarrow t_2(b^{-1}a)^{\pm 1}, & b^{\pm 1}t_2 \longrightarrow t_2(ba^{-1}b)^{\pm 1}, \\
t_1^{\pm 1}t^{-1} \longrightarrow t^{-1}(t_2^{-1}t_1)^{\pm 1}, & t_2^{\pm 1}t^{-1} \longrightarrow t^{-1}t_1^{\pm 1}, \\
b^{\pm 1}t^{-1} \longrightarrow t^{-1}b^{\pm 1}, & a^{\pm 1}t^{-1} \longrightarrow t^{-1}(b^{-1}a)^{\pm 1}, \\
b^{\pm 1}t_1^{-1} \longrightarrow t_1^{-1}a^{\pm 1}, & a^{\pm 1}t_1^{-1} \longrightarrow t_1^{-1}(a^2b^{-1}a)^{\pm 1}, \\
b^{\pm 1}t_2^{-1} \longrightarrow t_2^{-1}(ba)^{\pm 1}, & a^{\pm 1}t_2^{-1} \longrightarrow t_2^{-1}(ba^2)^{\pm 1}, \\
xx^{-1} \longrightarrow 1, & x^{-1}x \longrightarrow 1,
\end{array}
$$

where $x \in \{a, b, t_1, t_2, t\}$.

As a particular case, we get the rewriting system for the braid group $\mathcal{B}_3$.

References

Bergman, G. (1978), 'An Algorithmical Criterion for the Solvability of Algebraic Systems of Equations (German)', *Adv. in Math.* **29**, 178–218.

Birman, J. (1974), *Braid groups and mapping class groups*, Vol. 82 of *Annals of Math. Studies*, Princeton University Press.

Birman, J., Ko, K.-H. Lee, S.-J. (1998), 'A new approach to the word and conjugacy problem in the braid groups ', *Adv. Math.* **139**, 322–353.

Bokut, L. (1966), 'On a property of the Boone groups', *Algebra i Logika Sem.* **5**(5), 5–23.

Bokut, L. (1967), 'On a property of the Boone groups. II', *Algebra i Logika Sem.* **6**(1), 15–24.

Bokut, L. (1972), 'Unsolvability of the word problem, and subalgebras of finitely presented Lie algebras', *Izv. Akad. Nauk SSSR Ser. Mat.* **36**, 1173–1219.

Bokut, L. (1976), 'Imbeddings into simple associative algebras', *Algebra i Logika* **15**, 117–142, 245.

Bokut, L. (1980), Malcev's problem and groups with a normal form, *in* 'Stud. Logic Foundations Math., Word problems, II (Conf. on Decision Problems in Algebra, Oxford, 1976)', Vol. 95, North-Holland, Amsterdam-New York, pp. 29–53. With the collaboration of D.J. Collins.

Bokut, L. Shiao, L.-S. (2001), 'Gröbner–Shirshov bases for Coxeter groups', *Comm. Algebra* **29**(9), 4305–4319.

Bokut, L. Shiao, L.-S. (2002), 'Gröbner–Shirshov bases for Novikov and Boon groups', *Algebra Colloquium* . (to appear).

Book, R. Otto, F. (1993), *String-rewriting systems*, Springer-Verlag, New York.

Buchberger, B. (1965), An Algorithm for Finding the Bases Elements of the Residue Class Ring Modulo a Zero Dimensional Polynomial Ideal (German), Phd thesis, Univ. of Innsbruck (Austria).

Buchberger, B. (1970), 'An Algorithmical Criterion for the Solvability of Algebraic Systems of Equations (German)', *Aequationes Mathematicae* **4**(3), 374–383.

Burde, G. Zieschang, H. (1985), *Knots*, de Gruyter.

Cavicchioli, A., Hegenbarth, F. Kim, A. (1998), 'A geometric study of Sieradski groups', *Algebra Colloquium* **5**(2), 203–217.

Cohn, P. (1966), 'Some remarks on the invariant basis property', *Topology* **5**, 215–228.

Dehornoy, P. (2000), *Braids and self-distributivity*, Vol. 192 of *Progress in Math.*, Birkhäuser.

Eisenbud, D. (1995), *Commutative algebra. With a view toward algebraic geometry*, Vol. 150 of *Graduate Texts in Mathematics*, Springer-Verlag, New York.

Elrifai, E. Morton, H. (1994), 'Algorithms for positive braids', *Quart. J. Math. Oxford* **45**, 479–497.

Garber, D., Kaplan, S. Teicher, M. (2002), 'A new algorithm for solving the word problem in braid groups', *Adv. Math.* **167**, 142–159.

Garside, F. (1969), 'The braid group and other groups', *Quart. J. Math. Oxford* **20**, 235–254.

Gorin, E. Lin, V. (1969), 'Algebraic equations with continuous coefficients and some problems in the algebraic theory of braids (Russian)', *Mat. Sbornik* **78**(4), 579–610.

Helling, H., Kim, A. Mennicke, J. (1998), 'A geometric study of Fibonacci groups', *J. Lie Theory* **8**, 1–23.

Hermiller, S. Meier, J. (1999), 'Artin groups, rewriting systems and three-manifolds', *J. Pure Appl. Algebra* **136**, 141–156.

Heyworth, A. (2000), Rewriting as a special cases of non-commutative Gröbner basis, *in* 'Computational and geomteric aspects of modern algebra (Edinburg, Lecture Note Series', Vol. 275, Cambridge Univ. Press, pp. 101–105.

Johnson, D. (1980), *Topics in the theory of group presentations*, Vol. 42 of *London Math. Soc. Lect. Notes Series*, Cambridge University Press.

Kim, A., Kim, Y. Vesnin, A. (2000), On a class of cyclically presented groups, *in* Y. Baik, D. Johnson A. Kim., eds, 'Groups Korea'98', de Gruyter, pp. 211–220.

Lyndon, R. Schupp, P. (1977), *Combinatorial group theory*, Vol. 89 of *Ergebnisse der Mathematik und ihrer Grenzgebiete*, Springer-Verlag.

Markov, A. (1945), Foundations of the algebraic theory of braids (Russian), *in* 'Trudy Steklov Mat. Inst.', Moscow, pp. 1–54.

Newman, M. (1942), 'On theories with a combinatorial definition of "equivalence"', *Ann. of Math.* **43**, 223–243.

Shirshov, A. (1962), 'Some algorithm problems for Lie algebras (Russian)', *Sibirsk. Mat. Z.* **3**, 292–296. Translated in ACM SIGSAM Bull. Communications in Computer Algebra **33** (1999), no. 2, 3–9.

Thurston, W. (1992), Finite state algorithm for the braid group, *in* D. Epstein al, eds, 'Words processing in groups', Jones and Barlett, Boston and London.

Ufnarovski, V. (1998), Introduction to noncommutative Gröbner theory, *in* B. Buchberger F. Winkler, eds, 'Gröbner bases and applications', Cambridge University Press, pp. 259–280.

Gröbner Bases Property on Elimination Ideals in Finite Group Theory

Miguel Angel Borges-Trenard[1], Hebert Pérez-Rosés[2] and Mijail Borges-Quintana[1]

[1] *Departamento de Matemática, Facultad de Matemática y Computación, Universidad de Oriente, Santiago de Cuba, Cuba*

[2] *Departamento de Computación, Facultad de Matemática y Computación, Universidad de Oriente, Santiago de Cuba, Cuba*

E-mail: mborges@mabt.uo.edu.cu, hp@hrpc.uo.edu.cu, mijail@mbq.uo.edu.cu

Abstract

The Gröbner bases property on elimination ideals is one of the nice attributes of those bases that allow them to be a powerful tool for solving a great variety of algorithmic problems. The purpose of this note is to show how that property can be used in order to solve three computational group theoretical problems: computing discrete logarithms, computing inverses, and change of generators.

KEYWORDS: Gröbner bases, elimination ideals, group presentations, discrete logarithm problem.

Introduction

The Gröbner bases property on elimination ideals allows to obtain immediately a Gröbner basis of an elimination ideal if a Gröbner basis of the ideal has been computed. It has been widely used, particularly in commutative polynomial rings. The reader can see in (Buchberger Winkler 1998), and other works quoted there, applications of that property in different settings. In (Borges Borges 1998), that property was generalized for ideals in free associative algebras over a field. Here we are going to show how that generalization can be used for solving three problems in finite group theory:

1. The Discrete Logarithm Problem (**DLP**), see Section 3.1.
2. **Given** a finite group presentation, with generating set A, and an element w of the group (expressed in terms of A):
   Find the inverse of w (Section 3.2).

3. **Given** a finite group presentation, with generating set A, and a finite subset S of the group (expressed in terms of A):
   Find (if they exist) formulas that express A in terms of S (Section 3.3).

Of course, these problems have not trivial solution; for example, the complexity of finding a solution to the **DLP** is the base for the design of many systems in Cryptography. That problem has been extensively studied on some groups; for instance, multiplicative groups of finite fields and the group of points on an elliptic curve defined over a finite field (see (Koblitz 1998) for an overview). The solution presented here assumes that the group is given by a presentation, which is not typical in the cases mentioned before; however, in (Borges et al. 2000) an algorithm is presented for computing a complete presentation of a finite group when it is given by a concrete representation that allows to multiply elements, e.g. a permutation group.
On the other hand, the solution of the third problem allows to recognize whether a set of elements of a given group generates the whole group, which is also a well known difficult problem.

This paper does not intend to be self-contained. We recommend (Mora 1994) for an introduction to noncommutative Gröbner Bases and (Book Otto 1993) for string rewriting system technology. However, in Section 1 we shortly review the Gröbner bases property on elimination ideals and, in Section 2, the adaptation of that property for group presentations. We hope that those sections can be understood in essence, even without a very specialized knowledge of the theories. Section 3 and Remark 2.2 are the main contributions of this work.

1. Gröbner bases property on elimination ideals

Let $X := \{x_1, \ldots, x_n\}$ be a finite alphabet, $\langle X\rangle$ the free monoid on X, and $K\langle X\rangle$ the free associative algebra over X and the field K. On the other hand, let $K\langle X\rangle/I$ be the residue class algebra of $K\langle X\rangle$ modulo the two-sided ideal I and let $[f]_I$ be the image of $f \in K\langle X\rangle$ in $K\langle X\rangle/I$. Let $F := \{f_1, \ldots, f_m\}$ be a finite subset of $K\langle X\rangle$ and let $Ideal(F)$ be the two-sided ideal of $K\langle X\rangle$ generated by F. Finally, let $K\langle [F]_I\rangle := K\langle [f_1]_I, \ldots, [f_m]_I\rangle$ be the K-subalgebra of $K\langle X\rangle/I$ generated by the image of F.

For the reader's convenience, we write here the definition of term ordering with the elimination property that is given in (Borges Borges 1998).

Definition 1.1: **(Term ordering with the elimination property)**
The term ordering $\prec$ has the elimination property at the position k ($k \in [1, n-1]$) if:

$$\text{For every } s, t \in \langle X\rangle,\ s \prec t \text{ and } t \in \langle X_k\rangle \text{ implies } s \in \langle X_k\rangle,$$

where X_k stands for the subset of X composed by the first k letters.

That definition is equivalent to the following one:

$$\text{For } j \in (k, n] \text{ and for } s \in \langle X_k\rangle,\ s \prec x_j.$$

The problem below was solved in (Borges Borges 1998):

PROBLEM 1.1: **(Algebra Membership Problem)**
Given H, F finite subsets of $K\langle X\rangle$, and $f \in K\langle X\rangle$:
Decide whether $[f]_I \in K\langle [F]_I\rangle$, where $I := Ideal(H)$.

The solution method, that solves the Algebra Membership Problem by means of an Ideal Membership Test, can be summarized in the following algorithm:

ALGORITHM 1.1: **(Deciding whether $[\mathbf{f}]_\mathbf{I} \in \mathbf{K}\langle[\mathbf{F}]_\mathbf{I}\rangle$)**

1. Introduce a set of new letters $Y := \{y_1, \ldots, y_m\}$ for tagging the elements of F.
2. Choose a term ordering $\prec$ on $\langle Y \cup X\rangle$, with the elimination property at the position m, so that $Y \prec X$ (i.e. for each $y \in Y$, and for every $x \in X$, $y \prec x$).
3. Compute the reduced Gröbner basis* of J with respect to $\prec$ ($rGb(J, \prec)$), where
$$J := Ideal(\{y_1 - f_1, \ldots, y_m - f_m\} \cup H) \subset K\langle Y \cup X\rangle.$$
4. Compute $h := Can(f, rGb(J, \prec))$, that is the canonical form† of f w.r.t. $rGb(J, \prec)$.
5. Then $[f]_I \in K\langle [F]_I\rangle$ if and only if there exists $q \in K\langle Y\rangle$ such that $h = q$; moreover, in such a case: $f([x_1]_I, \ldots, [x_n]_I) = q([f_1]_I, \ldots, [f_m]_I)$.

A straightforward consequence of the previous algorithm is the following one.

ALGORITHM 1.2: **(Deciding whether $[\mathbf{f}]_\mathbf{I} \in \mathbf{K}\langle[\mathbf{F}]_\mathbf{I}\rangle$: An alternative)**

1. Introduce a set of new letters $Y := \{y_1, \ldots, y_m\}$ for tagging the elements of F and z for referring to f.
2. Choose a term ordering $\prec$ on $\langle Y \cup \{z\} \cup X\rangle$, with the elimination property at the position m, so that $Y \prec \{z\} \cup X$.
3. Compute the reduced Gröbner basis of J with respect to $\prec$, where
$$J := Ideal(\{y_1 - f_1, \ldots, y_m - f_m\} \cup \{z - f\} \cup H) \subset K\langle Y \cup \{z\} \cup X\rangle.$$
4. Then $[f]_I \in K\langle [F]_I\rangle$ iff there exists $q \in K\langle Y\rangle$ such that $z - q \in rGb(J, \prec)$.

*In fact, it is not mandatory to use the rGb, any G-basis would be enough.

†also called normal form.

Remark 1.1:

1. To call the two procedures given above "algorithms", one has to be sure that the reduced Gröbner basis is finite; it would be the case, for instance, when the ideal is zero-dimensional.
2. One has to introduce another letter in Algorithm 1.2; however, as a reward, the execution of the algorithm could be stopped as soon as a polynomial appears having the form $z - q'$, where $q' \in K\langle Y\rangle$. It is necessary to wait until the end of the algorithm just if the interest were in the "minimal" polynomial q such that $z - q$ is in the G-basis and $q \in K\langle Y\rangle$.

Another problem studied in (Borges Borges 1998) was the following one:

Problem 1.2: **(Algebras Equality Problem)**
Given F, H, and f as in Problem 1.1:
Decide whether $K\langle [F]_I\rangle = K\langle X\rangle/I$,

which can be solved in a way similar to the problem above. Summarizing: $K\langle [F]_I\rangle = K\langle X\rangle/I$ if and only if, for each $x_i \in X$, there exists $q_i \in K\langle Y\rangle$ such that

$$rGb(J, \prec) \setminus K\langle Y\rangle = \{x_1 - q_1, \ldots, x_n - q_n\}.$$

2. Particular case of group presentations

Let $\langle A \mid \sigma\rangle$ be a group presentation, then one can obtain at once a monoid presentation of the same group by setting $X := A \cup A^{-1}$ and adding to σ the trivial relations that define the inverses[‡].

There is a clear connection between congruences generated by subsets of $\langle X\rangle \times \langle X\rangle$ and ideals generated by certain kind of binomials in $K\langle X\rangle$. That relation is essential for interactions between the theories of Gröbner bases on the one side and String Rewriting Systems (SR-Systems) on the other side; namely:

$$\forall\ \sigma \subset \langle X\rangle \times \langle X\rangle \quad (s,t) \in \langle\sigma\rangle \Longleftrightarrow s - t \in Ideal(P(\sigma)), \tag{1}$$

where $\langle\sigma\rangle$ denotes the congruence generated by σ and $P(\sigma) := \{s - t \mid (s,t) \in \sigma\}$.

In particular, any Gröbner basis G of $Ideal(P(\sigma))$ is composed by binomials of the form mentioned above and, if one goes back by means of the set $\Sigma(G) := \{(s,t) \mid s - t \in G\}$, then another presentation of the same group is obtained; in addition, $\Sigma(G)$ is a complete SR-System (see, for instance, (Borges Borges 1998) and other references cited there).

For the sake of simplicity, we will denote by the same symbol a word $s \in \langle X\rangle$ and its image in some quotient ($[s]_I$ or, what is the same, $[s]_\sigma$). As usual, it will be clear from the context when we are referring to the word or its image.

[‡]We will also denote by σ the new set.

Some people could ask, why one has to use Gröbner bases where SR-Systems are enough? Well, a possible answer (not the only one) could be that the link between those theories allows to apply results from one of them in the framework of the other one. That is the case -for example- of the present work, where we show how the powerful elimination property of the Gröbner bases can be used in finite group theory.

Remark 2.1: One crucial point for applying results of Section 1 to the context of Section 3 is that the normal form of a word $s \in \langle X \rangle$, modulo a Gröbner basis of $Ideal(P(\sigma))$, is also a word (cf. (Borges Borges 1998)). With this observation in mind, the problems of Section 3 can be easily solved by means of specializations of Algorithms 1.1 and 1.2, which shows the strength of the method.

Remark 2.2: It is also worthy of consideration the possibility of simplifying the Gröbner basis associated with the group. It is based on the following result of (Borges Borges 1998):

Proposition 2.1: Let us suppose that $a^{-1} - w \in Ideal(P(\sigma))$, where $a \in A$ and $w \in \langle A \cup A^{-1} \setminus \{a^{-1}\} \rangle$, let $\prec$ be a term ordering on $\langle A \cup A^{-1} \rangle$ for which $w \prec a^{-1}$, and let β be obtained from σ by the replacement for w whenever a^{-1} appears. Then[§]:

$$rGb(P(\beta), \prec) = rGb(P(\sigma), \prec) \setminus \{a^{-1} - Can(w, rGb(P(\sigma), \prec))\},$$

where $rGb(P(\beta), \prec)$ stands for the reduced Gröbner basis of the ideal of $K\langle A \cup A^{-1} \setminus \{a^{-1}\} \rangle$ generated by $P(\beta)$.

Consequently, if a term ordering with the elimination property is chosen in such a way that $A \prec A^{-1}$, and the group is finite, then it can be possible to eliminate all the inverses (and their relations) from the Gröbner basis $rGb(P(\sigma), \prec)$ and obtain a set that characterizes the group as a monoid generated by A. We will denote this set by $\mathcal{G}(\sigma, \prec)$. To compute it will be the first step in the solution of the problems studied in this paper. This method leads to a complete monoid presentation[¶] that represents the group in a simpler way. Now we will exhibit a detailed example.

Example 2.1: **(Simplifying G-bases associated with finite groups)**

1. Let $\langle a | \sigma \rangle$ be a group presentation, where $\sigma := \{a^3 = a^2\}$. Then, taking into consideration the inverse of a, we get a monoid presentation of that group, with generating set $\{a, a^{-1}\}$ and set of defining relations $\sigma := \{a^3 = a^2, aa^{-1} = 1, a^{-1}a = 1\}$. It is easy to obtain that $rGb(P(\sigma)) = \{a - 1, a^{-1} - 1\}$, hence, $\mathcal{G}(\sigma, \prec) = \{a - 1\}$.

[§] $rGb(P(\gamma))$ represents, as usual, $rGb(Ideal(P(\gamma)))$.

[¶] $\Sigma(\mathcal{G}(\sigma, \prec))$.

2. Let us now consider the group A_4 given by $\sigma := \{a_1^3 = 1, a_2^3 = 1, a_1a_2a_1a_2 = 1\}$. Then, after adding the inverses (and their relations) and computing the corresponding Gröbner basis, we obtain:

$$rGb(P(\sigma), \prec) = \{a_1^3 - 1, a_2^2 - a_1a_2a_1, a_2a_1a_2 - a_1^2, a_2a_1^2a_2 - a_1^2a_2a_1^2,$$

$$a_1^{-1} - a_1^2, a_2^{-1} - a_1a_2a_1\}.$$

Therefore, $\mathcal{G}(\sigma, \prec)$ is formed by the relations in the first line above. This set will be the input in the examples of Section 3.
The reader can see, at the beginning of the next section, some details about the Alternating Group presentation and the term ordering that we have used.

3. Three problems

From now on, we consider $\langle A \mid \sigma \rangle$ to be a finite presentation of a finite group (where $A := \{a_1, \ldots, a_n\}$). Those finiteness conditions guarantee us that we can compute finite Gröbner bases of $Ideal(P(\sigma))$ (see Remark 1.1, item 1).

The examples of this section have been computed by using an implementation in **GAP** (Schönert et al. 1995) of Mora's procedure for computing Gröbner bases. The package was implemented in (Castellanos-Garzón 2002) and is available upon request to the first author of this paper.

The term ordering that we have used, with the elimination property, can be seen in (Borges Borges 1998). The selected group is A_n, which has the following presentation (Coxeter Moser 1972):

$$\sigma := \{a_i^3 = 1,\ (a_ia_j)^2 = 1 \ \mid \ 1 \leq i < j \leq n-2\}.$$

A Gröbner basis for $Ideal(P(\sigma))$ has been found in (Castellanos-Garzón 2002). We are going to work mainly with the particular case $n := 4$.

3.1. Discrete Logarithm Problem for finite groups

Problem 3.1: **(DLP to the base b)**
Given: $\langle A \mid \sigma \rangle$, $b \in \langle A \rangle$ (base element), and $c \in \langle A \rangle$:
Find: a positive integer k such that $b^k = c$ (if such k does exist).

In fact, our method -for solving the problem above- solves first the associated decision problem.

Algorithm 3.1: **(Solution to the Problem 3.1)**

1. Do in Algorithm 1.1: $F := \{b\}$, $H := \mathcal{G}(\sigma, \prec)$, $f := c$.
2. Then, k exists if and only if $Can(c, rGb(J, \prec)) \in \langle b \rangle$ and, if so, k is the positive integer that satisfies the relation $Can(c, rGb(J, \prec)) = b^k$. □

Of course, there are infinite possibilities for k, but the method leads to the minimal one, because normal forms are minimal with respect to the term ordering.

EXAMPLE 3.1: Let us set

$$\sigma := \{a_1^3 = 1, a_2^3 = 1, a_1a_2a_1a_2 = 1\},\ b := a_2a_1a_2,\ c := a_2a_1a_2a_2a_1a_2.$$

It is obvious that this example has solution, our intention is that the reader can understand easily and verify the result by himself. The set $\mathcal{G}(\sigma, \prec)$ was already built in Example 2.1. The **GAP** session leads immediately to the following reduced Gröbner basis:

$$rGb(J, \prec) := \{b^3 - 1, a_1 - b^2, a_2^2 - a_1a_2a_1, a_2ba_2 - ba_2b, a_2b^2a_2 - b\}.$$

Now, it is possible to check by hand that $Can(a_2a_1a_2a_2a_1a_2, rGb(J, \prec)) = b^2$.

3.2. Finding of the inverse

PROBLEM 3.2:
Given: $\langle A \mid \sigma\rangle,\ w \in \langle A\rangle$:
Find: $s \in \langle A\rangle$ so that $s = w^{-1}$.

ALGORITHM 3.2: **(Solution to the Problem 3.2)**

1. Introduce a new letter y for representing the inverse of w.
2. Choose a term ordering $\prec$ on $\langle A \cup \{y\}\rangle$ with the elimination property so that $A \prec y$.
3. Compute $rGb(J, \prec)$, where
$$J := Ideal(\mathcal{G}(\sigma, \prec) \cup \{yw - 1, wy - 1\}) \subset K\langle A \cup \{y\}\rangle.$$
4. Get the binomial of $rGb(J, \prec)$ whose maximal term is y.
5. Then s is the second term of that binomial.

In regard to the foundation of this algorithm, we can proceed as follows: It is clear that the relations $\{yw-1, wy-1\}$ do not alter the group under study and define the inverse of w. As the group is finite, there exists $t \in \langle A\rangle$ such that y and t are equal words in the group, thus, $y - t \in J$ (see equivalence 1 in Section 2). Consequently, if one takes into consideration that we are using a term ordering with the elimination property (compare Definition 1.1 and the remark below that definition) then one can be sure that $y - Can(t, rGb(J, \prec)) \in rGb(J, \prec)$. □

EXAMPLE 3.2: Let us take σ as in the Example 3.1, and $w := a_2a_1a_2$. Then:

$$rGb(J, \prec) := \{a_1^3 - 1, a_2^2 - a_1a_2a_1, a_2a_1a_2 - a_1^2, a_2a_1^2a_2 - a_1^2a_2a_1^2, y - a_1\};$$

therefore, $s := a_1$.

3.3. Conversion Formulas

PROBLEM 3.3:
Given: $\langle A \mid \sigma \rangle,\ S := \{s_1, \ldots, s_m\} \subset \langle A \rangle$:
Decide: whether S generates the given group and, in the affirmative case, compute $\{w_1, \ldots, w_n\} \subset \langle S \rangle$ such that, for $i \in [1, n],\ a_i = w_i$.

This situation can be interpreted as a particular case of the one given in Problem 1.2, where one also has to take into account Remark 2.1.

EXAMPLE 3.3: Let σ be as in the previous examples and let $s_1 := a_1^2 a_2 a_1, s_2 := a_1^2 a_2 a_1^2$. Now we are going to apply the first three steps of Algorithm 1.1 by taking $H := \mathcal{G}(\sigma, \prec)$, and $F := \{a_1^2 a_2 a_1, a_1^2 a_2 a_1^2\}$. Hence:

$$rGb(J, \prec) := \{s_1^3 - 1, s_2^2 - 1, s_2 s_1 s_2 - s_1^2 s_2 s_1^2, s_2 s_1^2 s_2 - s_1 s_2 s_1,$$

$$a_1 - s_1^2 s_2, a_2 - s_1 s_2\};$$

therefore[‖], $rGb(J, \prec) \setminus K\langle S \rangle = \{a_1 - s_1^2 s_2, a_2 - s_1 s_2\}$.
Consequently, the method responds affirmatively.

REMARK 3.1: **Experiment in GAP**

We have computed the same example on the more complicated group A_6.
The general form of $\mathcal{G}(\sigma, \prec)$, for the presentation of A_n given above, has been obtained in (Castellanos-Garzón 2002). For $n := 6$, $\mathcal{G}(\sigma, \prec)$ has 23 binomials. It took 14 minutes to compute the new Gröbner basis, which had 21 binomials. The relations between the sets $\{a_1, a_2\}$ and $\{s_1, s_2\}$ were kept in A_6, which was expected. Computation was made on a small Pentium of 300 MHz.

Conclusion

This paper contributes to show that the strength of the Gröbner bases property on elimination ideals can be successfully extended to non-commutative algebras. We also intend to encourage the reader to find new applications of that property in the context of Group Theory. Particularly, it would be interesting to find other group classes, different from the finite groups, where it could be possible to deal with problems similar to the ones exhibited here; it is strongly related to the finiteness of Gröbner bases.

[‖]See Problem 1.2.

Acknowledgement

The authors are indebted to Bruno Buchberger for ever. Particularly, Borges-Trenard and Borges-Quintana (father and son) discussed their Doctoral Thesis on Gröbner bases Theory with him and received Buchberger's support at every time and, which is more important, his influence as a very remarkable scientist and wonderful person.
Our gratitude to anonymous referees, their valuable suggestions contributed to improve this paper.

References

Book, R. V. Otto, F., eds (1993), *String Rewriting Systems*, Texts and Monographs in Computer Science, Berlin: Springer-Verlag.

Borges, M. A. Borges, M. (1998), *Gröbner Bases Property on Elimination Ideal in the Noncommutative Case*, Cambridge University Press, pp. 323–337 in (Buchberger Winkler 1998).

Borges, M. A., Borges, M. Mora, T. (2000), 'Computing Gröbner Bases by FGLM Techniques in a Non-commutative Setting', *Journal of Symbolic Computation* **30**(4), 429–449.

Buchberger, B. Winkler, F., eds (1998), *Gröbner Bases and Applications*, Vol. 251 of *London Mathematical Society Series*, Cambridge University Press. Proc. of the International Conference "33 Years of Gröbner Bases".

Castellanos-Garzón, J. A. (2002), On the use of GAP for computing Gröbner Bases (Spanish), Master thesis, Univ. of Oriente (Cuba).

Coxeter, H. S. M. Moser, W. O. J., eds (1972), *Generators and Relators for Discrete Groups*, Springer-Verlag, third edn.

Koblitz, N., ed. (1998), *Algebraic Aspects of Cryptography*, Vol. 3 of *Algorithms and Computation in Mathematics*, Springer-Verlag, New York.

Mora, T. (1994), 'An Introduction to Commutative and Noncommutative Gröbner Bases', *Theoretical Computer Science* (134), 131–173.

Schönert, M. et al. (1995), *GAP – Groups, Algorithms, and Programming*, fifth edn, Lehrstuhl D für Mathematik, Rheinisch Westfälische Technische Hochschule, Aachen, Germany.

Hilbert Polynomials in Two Variables and Bifiltered Ideals *

GIUSEPPA CARRA' FERRO

Department of Mathematics and Computer Science, University of Catania, Viale Andrea Doria 6, 95125 Catania, Italy

Abstract

In this paper it is shown a generalization of the Buchberger's algorithm, that allows to find a Levin's characteristic set of a bifiltered and bigraded ideal in a polynomial ring by using the notion of L-reduction. Such characteristic set allows to find the Hilbert polynomial in two variables of the corresponding bifiltered and bigraded quotient algebra of polynomials.

KEYWORDS: bidegree preserving term ordering, L-reduction, characteristic set, bivariate Hilbert polynomials

1. Introduction

Bifiltered and bigraded rings R, bifiltered and bigraded ideals and R-modules are useful tools for the study of Segre products of K-algebras, tensor products of graded algebras, Rees rings and symmetric algebras associated to homogeneous ideals in graded rings (blow-up algebras), affine and projective elimination theory, Weyl algebras and linear partial differential equations with coefficients in a polynomial ring.

Recently (Levin 1999) has studied Hilbert polynomials in two variables of a bifiltered submodule of a free module of finite rank over a bifiltered commutative ring of polynomials with coefficients in a field K, while (Robbiano Valla 1998) have studied Hilbert-Poincare' series in two variables of a bigraded K-subalgebra of a bigraded commutative ring of polynomials with coefficients in a field K.

Levin has shown that given a bifiltered submodule M of a free module E of finite rank p over a bifiltered commutative ring of polynomials $K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$ with coefficients in a field K and a set of generators $\{f_1, \ldots, f_r\}$ of the relation module $R(M)$ of M it is possible to find a characteristic set G of $R(M)$, e.g. another set of generators $G=\{g_1, \ldots, g_s\}$ of $R(M)$, such that the

*1991 Mathematics Subject Classification 13P99.

Hilbert polynomial in two variables of M is the numerical polynomial in two variables associated to the set of pairs of maximal terms $\{(u_{g_1}, v_{g_1}), \ldots, (u_{g_s}, v_{g_s})\}$ with respect to two term orderings, that are respectively degree preserving with respect to the sets of variables $\{X_1, \ldots, X_m\}$ and $\{Y_1, \ldots, Y_n\}$. Definitions and results in (Levin 1999) are different from the ones in (Robbiano Valla 1998).

The existence of such set of generators is proved by using the Ritt's characteristic set theory as in (Ritt 1950) and a notion of reduction, which is similar to the one used in Gröbner bases theory as in (Buchberger 1976), while the invariants of such polynomial with respect to the excellent bifiltrations are used for studying some properties of the corresponding linear systems of PDE's.

Let σ_X and σ_Y be term orderings on the set of terms in $K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$, that are respectively degree preserving with respect the sets of variables $\{X_1, \ldots, X_m\}$ and $\{Y_1, \ldots, Y_n\}$, and let I be an ideal with an excellent bifiltration in the bifiltered ring $K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$. The author shows an algorithm for a characteristic set $G=\{g_1, \ldots, g_s\}$ of I, that allows to find the Hilbert polynomial in two variables of the bifiltered and bigraded K-algebra $K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]/I$.

This algorithm can be easily extended to the case of bifiltered submodules of free modules of finite rank p over bifiltered rings of polynomials.

2. Preliminaries

Let K be a field and let $R=K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$ be the polynomial ring in $m+n$ variables with coefficients in K. Let $\mathbf{N}_0 = \{0, 1, \ldots, n, \ldots\}$.

Definition: $T_X=\{\ X_1^{a_1} \cdots X_m^{a_m}$: $(a_1, \ldots, a_m) \in \mathbf{N}_0^m\ \}$ is the set of terms in R in the variables $X_1, \ldots, X_m$.
$T_Y=\{\ Y_1^{b_1} \cdots Y_n^{b_n}$: $(b_1, \ldots, b_n) \in \mathbf{N}_0^n\ \}$ is the set of terms in R in the variables $Y_1, \ldots, Y_n$.
$T_{XY}=\{\ X_1^{a_1} \cdots X_m^{a_m} Y_1^{b_1} \cdots Y_n^{b_n}$: $(a_1, \ldots, a_m, b_1, \ldots, b_n) \in \mathbf{N}_0^{m+n}\ \}$ is the set of terms in R in the variables $X_1, \ldots, X_m, Y_1, \ldots, Y_n$.

T_{XY} is a monoid and it is the product of the monoids T_X and T_Y.

If $t \in T_{XY}$, then $deg_X t = \sum_{i=1,\ldots,m} a_i$, $deg_Y t = \sum_{j=1,\ldots,n} b_j$, while $deg(t) = \sum_{i=1,\ldots,m} a_i + \sum_{j=1,\ldots,n} b_j$.

If $r, s \in \mathbf{Z}$, then $T(r,s)=\{t \in T_{XY}$: $deg_X t \leq r$ and $deg_Y t \leq s\}$.

2.1. Bifiltered and Bigraded Rings

Here some basic properties of bifiltered and bigraded rings are introduced.

Definition: Let $R=K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$. A bisequence $(R_{rs})_{r,s \in \mathbf{Z}}$ of vector K-subspaces of R is called a *bifiltration of* R if:
(i) $R_{rs} = 0$ if either r or s is negative.

(ii) $\cup\{R_{rs} : r, s \in \mathbf{Z}\}$=$R$.
(iii) $R_{rs} \subseteq R_{r+1,s}$ and $R_{rs} \subseteq R_{r,s+1}$ for all r and s.
(iv) $R_{rs}R_{hk} = R_{r+h,s+k}$ for all r, s, h, k.

R=$K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$ is bifiltered by taking R_{rs} equal to the vector K-space generated by $T(r, s)$.

Definition: Let R=$K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$ and let M be a R-module. A bisequence $(M_{rs})_{r,s\in\mathbf{Z}}$ of vector K-subspaces of M is called a *bifiltration of* M if:
(i) $M_{rs} = 0$ if either $r \leq r_0$ or $s \leq s_0$ and $\cup\{M_{rs} : r, s \in \mathbf{Z}\}$=$M$.
(ii) $M_{rs} \subseteq M_{r+1,s}$ and $M_{rs} \subseteq M_{r,s+1}$ for all r and s.
(iii) $R_{rs}M_{hk} \subseteq M_{r+h,s+k}$ for all r, s, h, k.
A bifiltration is called *excellent* if :
(iv) M_{rs} is a finitely generated vector K-space for all r ed s.
(v) $R_{rs}M_{hk} = M_{r+h,s+k}$ for all $r \geq r^*$, $s \geq s^*$ and all h and k nonnegative.

An ideal I of R is bifiltered if it is bifiltered as R-module. If I=$(f_1, \ldots, f_p)$, then $(I_{rs}$=$\sum_{i=1,\ldots,p} R_{rs}f_i)_{(r,s)\in\mathbf{Z}^2}$ is a excellent bifiltration of I.

Definition: Let R=$K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$. A bisequence $(R'_{rs})_{r,s\in\mathbf{Z}}$ of vector K-subspaces of R is called a *bigraduation of* R if:
(i) $R'_{rs} = 0$ if either r or s is negative.
(ii) $\bigoplus\{R'_{rs} : r, s \in \mathbf{Z}\}$=$R$.
(iii) $R'_{rs}R'_{hk} \subseteq R'_{r+h,s+k}$ for all r, s, h, k.

R=$K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$ is bigraded by taking R'_{rs} equal to the vector K-space generated by $T'(r, s)$=$\{t \in T(r, s)$: $deg_X t = r$ and $deg_Y t = s$ $\}$.

If R is bigraded with the bigraduation $(R'_{rs})_{(r,s)\in\mathbf{Z}^2}$, then R is bifiltered with the bifiltration $(R_{rs})_{(r,s)\in\mathbf{Z}^2}$ defined by R_{rs}= $\bigoplus\{R'_{hk} : h \leq r, k \leq s\}$.

If R is bifiltered with the bifiltration $(R_{rs})_{(r,s)\in\mathbf{Z}^2}$, then R is bigraded with the bigraduation $(R'_{rs})_{(r,s)\in\mathbf{Z}^2}$, where R'_{rs}=$R_{rs}/(R_{r-1,s} \cap R_{r,s-1})$.

2.2. Term Orderings

Here some basic properties of bidegree preserving term orderings are introduced.

Definition: A term ordering σ on T_{XY} is a total order such that:

(i). $1 <_\sigma t$ for all $t \in T_{XY} \setminus \{1\}$;

(ii). $t_1 <_\sigma t_2$ implies $t_1t' <_\sigma t_2t'$ for all $t' \in T_{XY}$.

Definition: (Levin 1999). A term ordering σ on the set of terms T_{XY} is *X-bidegree preserving* if $t_1 <_\sigma t_2$ when either $deg_X t_1 < deg_X t_2$ or $deg_X t_1 = deg_X t_2$ and $deg_Y t_1 < deg_Y t_2$.
A term ordering σ on the set of terms T_{XY} is *Y-bidegree preserving* if $t_1 <_\sigma t_2$ when either $deg_Y t_1 < deg_Y t_2$ or $deg_Y t_1 = deg_Y t_2$ and $deg_X t_1 < deg_X t_2$.

EXAMPLE 1: *Let* $t_1, t_2 \in T_{XY}$. *Let* $t_1 = X_1^{a_1} \cdots X_m^{a_m} Y_1^{b_1} \cdots Y_n^{b_n}$ *and let* $t_2 = X_1^{c_1} \cdots X_m^{c_m} Y_1^{d_1} \cdots Y_n^{d_n}$. *If* σ *is defined by* $t_1 <_\sigma t_2$ *if either* $(\sum_{i=1,\ldots,m} a_i, \sum_{j=1,\ldots,n} b_j) <_{lex} (\sum_{i=1,\ldots,m} c_i, \sum_{j=1,\ldots,n} d_j)$ *or* $(\sum_{i=1,\ldots,m} a_i, \sum_{j=1,\ldots,n} b_j) = (\sum_{i=1,\ldots,m} c_i, \sum_{j=1,\ldots,n} d_j)$ *and* $(a_1, \ldots, a_m, b_1, \ldots, b_n) <_\tau (c_1, \ldots, c_m, d_1, \ldots, d_n)$, *where* τ *is a term ordering on* T_{XY}, *then* σ *is a* X*-bidegree preserving term ordering on* T_{XY}. *In similar way if* σ *is defined by* $t_1 <_\sigma t_2$ *if either* $(\sum_{j=1,\ldots,n} b_j, \sum_{i=1,\ldots,m} a_i) <_{lex} (\sum_{j=1,\ldots,n} d_j, \sum_{i=1,\ldots,m} c_i)$ *or* $(\sum_{j=1,\ldots,n} b_j, \sum_{i=1,\ldots,m} a_i) = (\sum_{j=1,\ldots,n} d_j, \sum_{i=1,\ldots,m} c_i)$ *and* $(b_1, \ldots, b_n, a_1, \ldots, a_m) <_\tau (d_1, \ldots, d_n, c_1, \ldots, c_m)$, *where* τ *is a term ordering on* T_{XY}, *then* σ *is a* Y*-bidegree preserving term ordering on* T_{XY}.

Now let σ_X and σ_Y be respectively a X-bidegree preserving term ordering and a Y-bidegree preserving term ordering.

If $f \in K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$, then $f = \sum_{h=1,\ldots,w} a_{ih} t_{ih} = \sum_{h=1,\ldots,w} a_{jh} t_{jh}$ with $t_{i1} >_{\sigma_X} t_{i2} >_{\sigma_X} \ldots >_{\sigma_X} t_{iw}$ and $t_{j1} >_{\sigma_Y} t_{j2} >_{\sigma_Y} \ldots >_{\sigma_Y} t_{jw}$ with nonzero $a_{ih} \in K$ and $t_{ih}, t_{jh} \in T_{XY}$ for all h.

$M_{\sigma_X}(f) = a_{i1} t_{i1}$ is the *leading* σ_X*-monomial of* f *with respect to* σ_X and $M_{\sigma_Y}(f) = a_{j1} t_{j1}$ is the *leading* σ_Y*-monomial of* f *with respect to* σ_Y.

$a_{i1} = lc_{\sigma_X}(f)$ and $a_{j1} = lc_{\sigma_Y}(f)$ are respectively the *leading coefficients of* f *with respect to* σ_X and σ_Y, while $t_{i1} = T_{\sigma_X}(f) = u_f$ and $t_{j1} = T_{\sigma_Y}(f) = v_f$ are the *leading terms of* f *with respect to* σ_X and σ_Y.

3. L-Reduction

Here the definitions and properties following from the notion of reduction given by Levin (1999) are introduced. Such reduction will be called *L-reduction.*

Definition: (Levin 1999). Let $f, g \in K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$ and let σ_X and σ_Y be respectively a X-bidegree preserving and a Y-bidegree preserving term ordering on T_{XY}. f is L-reduced with respect to g if f does not contain any term tu_g, such that $deg_Y tv_g \leq deg_Y v_f$.
A subset F of $R = K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$ is L-autoreduced if each element of F is L-reduced with respect to the other ones.

EXAMPLE 2: *Let* $R = K[X_1, Y_1]$. *Let* σ_X *and* σ_Y *be as above with* τ*=lexicographic. Let* $f = X_1^2 - Y_1$ *and* $g = X_1 - Y_1^2$. $u_f = X_1^2$, $v_f = Y_1$, $u_g = X_1$ *and* $v_g = Y_1^2$. $\{f, g\}$ *is a L-autoreduced set.*

In (Levin 1999) it is shown that each L-autoreduced subset F of R is finite and each $f \in R$ can be L-reduced in a finite number of steps with respect to an L-autoreduced subset F=$\{f_1, \ldots, f_r\}$ of R. Moreover f L-reduces to a polynomial g with respect to F and there exist $g_1, \ldots, g_r \in R$, such that $f = \sum_{i=1,\ldots,r} g_i f_i + g$

and g is L-reduced with respect to F. g is called a *L-normal form of f with respect to F*.

L-REDUCTION ALGORITHM (Levin)
Input $f \in R$, a positive integer r, $F = \{f_1, \ldots, f_r\}$, where $f_i \neq 0$ for all $i = 1, \ldots, r$.
Output $g \in R$ and $g_1, \ldots, g_r \in R$, such that $f = \sum_{i=1,\ldots,r} g_i f_i + g$ and g is L-reduced with respect to F.
Begin
$g_1 := 0, \ldots, g_r := 0$, $g := f$
While there exists i, $i = 1, \ldots, r$, and a term t, that appears in g with a nonzero coefficient a_t, such that u_{f_i} divides t and $deg_Y(\frac{t}{u_{f_i}} v_{f_i}) \leq deg_Y v_f$ **do**
z:=the greatest (with respect to σ_X) of the terms t, that satisfy the above conditions.
j:=the smallest number i for which u_{f_i} is the greatest (with respect to σ_X) leading σ_X-monomial of an element f_i, such that u_{f_i} divides z and $deg_Y(\frac{z}{u_{f_i}} v_{f_i}) \leq deg_Y v_g$.
$g_j := g_j + \frac{a_z z}{lc_{\sigma_X}(f_j) u_{f_j}} f_j$ and $g := g - \frac{a_z z}{lc_{\sigma_X}(f_j) u_{f_j}} f_j$.
End

Remark: The definition of L-reduction given by Levin is weaker than the one used in Gröbner bases theory (Buchberger 1976). In fact given a X-bidegree preserving term ordering if f is reduced with respect to g, then it is L-reduced. The converse it is not true as in the above example.

In (Levin 1999) it is given also the definition of *ranking* on the set of all polynomials, which is a pre-order, and the definition of *ranking on the sets* of all L-autoreduced subsets of polynomials.

Definition: (Levin 1999). Let $f, g \in K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$ and let σ_X and σ_Y be respectively a X-bidegree preserving and a Y-bidegree preserving term ordering on T_{XY}. f has lower rank than g and write $rk(f) < rk(g)$ if either $u_f <_{\sigma_X} u_g$ or $u_f = u_g$ and $v_f <_{\sigma_Y} v_g$. If $u_f = u_g$ and $v_f = v_g$ then f and g have the same rank and write $rk(f) = rk(g)$.

Let F=$\{f_1, \ldots, f_r\}$ be a subset of R. We always suppose that $rank(f_1) \leq rank(f_2) \leq \ldots \leq rank(f_r)$.

Definition: (Levin 1999). Let F=$\{f_1, \ldots, f_r\}$ and G=$\{g_1, \ldots, g_s\}$ be L-autoreduced subsets of R. F has rank lower than G if one of the following cases holds:
(1) there exists $h \in \mathbf{N}_0$ such that $h \leq min\{r, s\}$, $rk(f_i) = rk(g_i)$ for all $i = 1, \ldots, h-1$ and $rk(f_h) < rk(g_h)$;
(2) $r > s$ and $rk(f_i) = rk(g_i)$ for all $i = 1, \ldots, s$.
If $r = s$ and $rk(f_i) = rk(g_i)$ for all $i = 1, \ldots, s$, then F has the same rank as G.

By using the same tools of Ritt's theory, Levin shows that each family of L-autoreduced sets has a minimal element with respect to the ranking. If the family of all L-autoreduced subsets of an ideal I of R is considered, then a L-autoreduced subset of I minimal with respect to the ranking is called a *characteristic set* of the ideal I. Finally Levin shows that the polynomials in the characteristic set are a set of generators of the ideal I. No algorithm is given in order to find such characteristic set.

Now the relation between Gröbner bases and characteristic set will be investigated.

Definition: Let I be an ideal in $K[X_1,\ldots,X_m,Y_1,\ldots,Y_n]$ and let σ be a term ordering on T_{XY}. $M_\sigma(I)$=($M_\sigma(f)$: $f \in I$). If $G \subseteq I$, then $M_\sigma(G)$=($M_\sigma(g)$: $g \in G$).

The following definition is well known (Buchberger 1976).

Definition: Let I be an ideal in $K[X_1,\ldots,X_m,Y_1,\ldots,Y_n]$ and let σ be a term ordering on T_{XY}. G is a Gröbner basis of I with respect to σ if and only if $M_\sigma(I)$=$M_\sigma(G)$.

The existence and the minimality of a characteristic set of an ideal I with respect to the ranking imply the following theorem.

THEOREM 3.1: *Let I be an ideal in $K[X_1,\ldots,X_m,Y_1,\ldots,Y_n]$ and let σ_X and σ_Y be respectively a X-bidegree preserving and a Y-bidegree preserving term ordering on T_{XY}. Let G=$\{g_1,\ldots,g_s\}$ be a characteristic set of I.*

(i). G is a Gröbner basis of I with respect to σ_X.

(ii). If f is a nonzero element of I and f=$\sum_{i=1,\ldots,r} a_i t_i g_{j(i)}$ where $a_i \in K$ with $a_i \neq 0$, $t_i \in T_{XY}$, $j(1) \in \{1,\ldots,s\}$ for all i by L-reduction algorithm, then $deg_Y v_f \geq deg_Y t_i v_{g_{j(i)}}$ for all $i = 1,\ldots,r$ and $deg_Y v_f = deg_Y t_i v_{g_{j(i)}}$ for some i.

(iii). Let d=$min\{deg_Y f$: $f \in I$, $f \neq 0\}$. For every $f \in I$ with $deg_Y f = d$ there exists at least one $g \in G$, such that u_g is a term of f and $deg_Y v_g = d$

(iv). Let $g \in G$ with $deg_Y g = d$. If F is a Gröbner basis of I with respect to σ_Y, then there exists $f' \in F$ with $deg_Y f' = d$, such that $g = af'$ for some nonzero $a \in K$.

Proof. (i). By (Levin 1999), thm.4.4. if $f \in I$, then f is L-reduced with respect to G if and only if f=0. Since $G \subset I$, then $M_{\sigma_X}(G) \subseteq M_{\sigma_X}(I)$. Now let f be a nonzero element of I. Since f L-reduces to zero with respect to G, then it reduces to zero with respect to G and with respect to σ_X. Therefore $M_{\sigma_X}(f)$=$lc_{\sigma_X}(f)u_f$=$rtM_{\sigma_X}(g)$ for some $r \in R$, $t \in T_{XY}$ and $g \in G$. It follows that $M_{\sigma_X}(I) \subseteq M_{\sigma_X}(G)$ and then (i) follows by definition 11.

(ii). Now let $G = \{g_1, \ldots, g_s\}$ with $rk(f_1) < \ldots < rk(f_s)$ and let f be a nonzero element of I. By L-reduction algorithm $f{=}\sum_{i=1,\ldots,r} a_i t_i g_{j(i)}$ where $a_i \in K$ with $a_i \neq 0$, $t_i \in T_{XY}$, $j(i) \in \{1, \ldots, s\}$ for all i and $t_1 u_{g_{j(1)}} \geq_{\sigma_X} \ldots \geq_{\sigma_X} t_r u_{g_{j(r)}}$. Furthermore $u_f = t_1 u_{g_{j(1)}}$. If $r = 1$, then $v_f = t_1 v_{g_{j(1)}}$ and (ii) is proved. Now suppose that $r > 1$. Let's suppose that $deg_Y v_f \geq deg_Y t_i v_{g_{j(i)}}$ for all $i = 1, \ldots, r'$ and $deg_Y v_f = deg_Y t_i v_{g_{j(i)}}$ for some i, whenever $f{=}\sum_{i=1,\ldots,r'} a_i t_i g_{j(i)}$ and $r' \leq r - 1$. Let $f_1 = f - a_1 t_1 g_1$. $f_1 \in I$ and by induction hypothesys $deg_Y v_{f_1} = deg_Y t_i v_{g_{j(i)}}$ for some $i = 2, \ldots, r$. Since f is L-reducible with respect to G, then $deg_Y v_f \geq deg_Y t_1 v_{g_{j(1)}}$. So either $v_{f_1} = v_f$, when $t_1 v_{g_{j(1)}} <_{\sigma_Y} v_f$ or $v_f = t_1 v_{g_{j(1)}}$ or $v_{f_1} = t_1 v_{g_{j(1)}}$, when $deg_Y v_f = deg_Y t_1 v_{g_{j(1)}}$ and $v_f <_{\sigma_Y} t_1 v_{g_{j(1)}}$. Now (ii) follows by induction hypothesys.

(iii). By (ii) if f is a nonzero element of I and $deg_Y f = d$, then f L-reduces to zero with respect to G. So there exists a $g \in G$, such that tu_g is a term of f and $d = deg_Y f = deg_Y v_f \geq deg_Y t v_g = deg_Y t + deg_Y v_g$. So $deg_Y t = 0$ and $deg_Y v_g = d$, e.g. $t \in T_X$.

(iv) If F is a Gröbner basis of I with respect to σ_Y and $f \in F$ with $deg_Y f = deg_Y v_f = d$, then by (iii) there exists at least one $g \in G$ with $deg_Y g = deg_Y v_g = d$. Since $g \in I$, then there exists at least one $f' \in F$ with $deg_Y f' = deg_Y v_{f'} = d$, such that $v_g = t' v_{f'}$ and $t' \in T_X$. If $lc_{\sigma_Y}(f')g - lc_{\sigma_X}(g)t'f' \neq 0$, then $deg_Y(lc_{\sigma_Y}(f')g - lc_{\sigma_Y}(g)t'f') < d$. So we have a contradiction by minimality of d. If $a = \frac{lc_{\sigma_Y}(g)}{lc_{\sigma_Y}(f)}$, then $g = at'f'$. By (iii) there exists $g' \in G$, such that f' contains a term $tu_{g'}$ with $t \in T_X$ and $deg_Y f' = deg_Y t g' = deg_Y t v_{g'} = deg_Y v_{g'} = d$. If $g' \neq g$, then g is not L-reduced with respect to g' and we have a contradiction by definition of characteristic set. So $g = g'$ and then $t = t' = 1$, e.g. $g = af'$.

Remark: If F is a reduced Gröbner basis of I with respect to σ_X, then it is a L-autoreduced subset of I by definition of L-reduction and $rank(F)$ is greater than or equal to $rank(G)$, where G is a characteristic set of I, by (i) of the theorem 1.

4. An Algorithm for the Characteristic Set

In this section it is presented an algorithm for the characteristic set of an ideal I in R with respect to the L-reduction.

Since Levin's theory is an extension of Ritt's theory, then one can try procedures analogous to Ritt's procedure (Ritt 1950) for partial differential equations by using Fourier transform. The usual Ritt's procedures find a characteristic set of a finite set and an extended characteristic set of an ideal. If only linear polynomials are considered, then such extended characteristic set is a characteristic set.

Furthermore the Ritt's notions of reduction and characteristic sets in the case of linear partial differential equations with constants coefficients coincide respectively with the notions of reduction in Gröbner bases theory and reduced Gröbner bases as in (Carra'-Ferro 2001, Kondratieva Levin Mikhalev Pankratiev

1999). So an algorithm similar to the Buchberger's algorithm for Gröbner bases with the new definition of L-reduction can be used in order to find such Levin's characteristic sets.

Definition: Let $f, g \in K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$ and let σ_X and σ_Y be respectively a X-bidegree preserving and a Y-bidegree preserving term ordering on T_{XY}. $S(f,g) = lc_{\sigma_X}(g)\frac{l}{T_{\sigma_X}(f)}f - lc_{\sigma_X}(f)\frac{l}{T_{\sigma_X}(g)}g = lc_{\sigma_X}(g)\frac{l}{u_f}f - lc_{\sigma_X}(f)\frac{l}{u_g}g$, where $l = l.c.m.(u_f, u_g)$.

BUCHBERGER'S ALGORITHM FOR L-REDUCTION

Input $F = \{f_1, \ldots, f_r\}$ a basis of the ideal I in R, a X-bidegree preserving term ordering σ_X and a Y-bidegree preserving term ordering σ_Y on T_{XY}.
Output $G = \{g_1, \ldots, g_s\}$ is a L-autoreduced basis of I.

$s := r$
$H := F$
$P := \{(i,j) : 1 \leq i < j \leq r\}$;
while P is nonempty
do Choose $(i,j) \in P$;
$Q := S(f_i, f_j)$;
$P := P \setminus \{(i,j)\}$;
L-reduce Q with respect to H;
if $Q \neq 0$ **then**
$P := P \cup \{(i, s+1) : 1 \leq i \leq s\}$;
$f_{s+1} := Q$;
$H := H \cup \{Q\}$;
$s := s + 1$;
return H;
begin $G = \emptyset$, $H = E$;
while $E \neq \emptyset$ **do**
select e_0 from E;
$E := E \setminus e_0$;
if e_0 is L-reduced with respect to all $e \in E$ **and**
e_0 is L-reduced with respect to all $g \in G$ **then**
$G := G \cup e_0$ **end**;
end

THEOREM 4.1: *Let $F = \{f_1, \ldots, f_r\} \subset K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$. Let $I = (f_1, \ldots, f_r)$ be an ideal in $K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$ and let σ_X and σ_Y be respectively a X-bidegree preserving and a Y-bidegree preserving term ordering on T_{XY}. Let F_1 be a Gröbner basis of I with respect to σ_X, given by the usual Buchberger algorithm with input F. Let F_2 be a Gröbner basis of I with respect to σ_Y, given by the usual Buchberger algorithm with input F_1. Let G be the output of the Buchberger algorithm for L-reduction with input F_2. G is a characteristic set of I.*

Proof. $F \subseteq F_1 \subseteq F_2$ by definition of Gröbner basis with respect to a term ordering and $I{=}(F){=}(F_1){=}(F_2)$. Moreover F_2 is a Gröbner basis of I with respect to σ_X and it contains every polynomial g in a characteristic set of I with $deg_Y v_g = d$ up to a nonzero element $a \in K$ by (iv) of theorem 1. Let H be as in the Buchberger algorithm for L-reduction with input $F := F_2$. $F_2 \subseteq H$ and then $I{=}(H)$, because each element h of H is in $I{=}(F_2)$ by its own definition. Let G be the output of the Buchberger algorithm for L-reduction with input $F := F_2$. $G \subseteq H$ so $(G) \subseteq I$. On the other hand every $h \in H \setminus G$ is in (G) by definition of L-reduction. It follows that $I = (H) \subseteq (G)$, e.g. $I{=}(G)$. G is L-autoreduced by its own definition. In order to prove that G is a characteristic set of I it is sufficient to prove that each nonzero $f \in I$ L-reduces to zero with respect to G by (Levin 1999), thm.4.4. . If $g, g' \in G$, then $S(g, g') \in H$ and it L-reduces to zero with respect to H, by definition of Buchberger algorithm for L-reduction. Since each $h \in H \setminus G$ L-reduces to zero with respect to G, then $S(g, g')$ L-reduces to zero with respect to G. By repeating the proof in (Becker Weispfenning 1993), lemma 5.44 p.210 and thm.5.48 p.211 $((iii) \Rightarrow (i))$ for L-reduction each $f \in I$ L-reduces to a unique element $f_0 \in I$, because the L-reduction is also a reduction with respect σ_X. Finally we have to show that $f_0 = 0$. Since $I{=}(G)$, then by repeating the proof in (Becker Weispfenning 1993), thm. 5.35 p.206 $((iv) \Rightarrow (v))$ and lemma 5.26 p.202) for L-reduction f and zero L-reduce to the same $f_0 \in I$. But zero L-reduces to zero and then $f_0 = 0$, because each $f \in I$ L-reduces to a unique element of I. Now G is a characteristic set by (Levin 1999), thm.4.4. .

EXAMPLE 3: *Let* $F{=}\{Y_1^2-1, X_1-Y_1\}$ *and let* $I{=}(F)$. σ_X*=lexicographic term ordering with* $Y_1 <_{\sigma_X} X_1$ *and* σ_Y*=lexicographic term ordering with* $X_1 <_{\sigma_Y} Y_1$. *F is a reduced Gröbner basis of I with respect to* σ_X *and then* $F = F_1$. $F_2{=}\{Y_1^2-1, X_1-Y_1, X_1^2-1\}$ *is the Gröbner basis of I with respect to* σ_Y *given by the Buchberger algorithm with input* F_1. $F_1 \subset F_2$. $G{=}\{Y_1^2-1, X_1-Y_1, X_1Y_1-1, X_1^2-1\}$, *because* $S(X_1-Y_1, X_1^2-1)$ *L-reduces to* X_1Y_1-1 *with respect to* F_2.

Remark: Example 4 shows that it is necessary to have a Gröbner basis of I with respect to σ_X and to σ_Y as input of the Buchberger's algorithm for L-reduction in order to find a characteristic set of I. In fact the Buchberger's algorithm for L-reduction with input $F := F = F_1$ has as output $G = F = F_1$, that is not a characteristic set of I.

5. Hilbert Polynomials in Two Variables

Here the notion of Hilbert polynomial in two variables given by Levin is introduced and some properties are shown.

THEOREM 5.1: (*Levin 1999*) *Let I be an ideal in* $R = K[X_1, \ldots, X_m, Y_1, \ldots, Y_n]$ *and let* σ_X *e* σ_Y *be respectively a X-bidegree preserving and a Y-bidegree preserving term ordering on* T_{XY}. *Let* $G = \{g_1, \ldots, g_k\}$ *be a characteristic set of I and let* $V = R/I$.

(i). *V is a bifiltered R-module with the bifiltration $(V_{rs})_{(r,s)\in\mathbf{Z}^2}$, where V_{rs} as vector K-space is generated by U_{rs}=$U'_{rs}\cup U''_{rs}$, where U'_{rs}= $\{t\in T$: $deg_X t\leq r$, $deg_Y t\leq s$ and t is not a multiple of any u_{g_j} $j=1,\ldots,k\}$ and U''_{rs}= $\{t\in T$: $deg_X t\leq r$, $deg_Y t\leq s$ e $t=wu_{g_j}$ with $deg_Y wv_{g_j}>s$ for all $w\in T_{XY}$ e $j=1,\ldots,k\}$.*

(ii). *$(V_{rs})_{(r,s)\in\mathbf{Z}^2}$ is an excellent bifiltration of V.*

THEOREM 5.2: (*Levin 1999*) *Let I be an ideal in $R=K[X_1,\ldots,X_m,Y_1,\ldots,Y_n]$ and let σ_X e σ_Y be respectively a X-bidegree preserving and a Y-bidegree preserving term ordering on T_{XY}. Let $G=\{g_1,\ldots,g_k\}$ be a characteristic set of I and let $V=R/I$. Then there exists a numerical polynomial $\omega_I(t_1,t_2)$ in two variables t_1 and t_2 such that:*

(i). *$\omega_I(t_1,t_2)=dim_K(V_{t_1t_2})$ for all $t_1\geq t_1^*$ and $t_2\geq t_2^*$.*

(ii). *$deg_{t_1}(\omega_I(t_1,t_2))\leq m$ and $deg_{t_2}(\omega_I(t_1,t_2))\leq n$, and $\omega_I(t_1,t_2))=\sum_{i=0,\ldots,m}\sum_{j=0,\ldots,n}a_{ij}\binom{t_1+i}{i}\binom{t_2+j}{j}$, where $a_{ij}\in\mathbf{Z}$ for all i and j.*

(iii). *Let A=$\{(i,j)$: $i=0,\ldots,m$, $j=0,\ldots,n$ and $a_{ij}\neq 0\}$, let $\mu=(\mu_1,\mu_2)$ ed $\nu=(\nu_1,\nu_2)$ the maximal elements in A respectively with respect to the lexicographic and reverse-lexicographic term orderings on $\mathbf{N}_0^2$. μ, ν, a_{mn}, $a_{\mu_1\mu_2}$, $a_{\nu_1\nu_2}$ do not depend on the excellent bifiltration of V=R/I.*

The *Hilbert polynomial in two variables of I* as in (Levin 1999) is the numerical polynomial in two variables $H_I(t_1,t_2)=\omega_I(t_1,t_2)=dim_K(V_{t_1t_2})=\omega_1(t_1,t_2)+\omega_2(t_1,t_2)$, where $\omega_1(t_1,t_2)=card(U'_{t_1,t_2})$ and $\omega_2(t_1,t_2)=card(U''_{t_1,t_2})$, when $t_1\geq t_1^*$ e $t_2\geq t_2^*$, while it is defined as $\omega_I(t_1,t_2)=card(U'_{t_1,t_2})$ in (Caboara DeDominicis Robbiano 1996, Robbiano Valla 1998).

The Hilbert polynomial in two variables can be found by using either algorithms in (Kondratieva Levin Mikhalev Pankratiev 1992, Levin 1999) or algorithms in (Caboara DeDominicis Robbiano 1996).

EXAMPLE 4: (*Levin 1999*). $I=(X_1+Y_1^2+1)$. $\omega_1(t_1,t_2)=\binom{t_1+1}{1}\binom{t_2+1}{1}-\binom{t_1}{1}\binom{t_2+1}{1}=t_2+1$. $\omega_2(t_1,t_2)=\binom{t_1}{1}(\binom{t_2+1}{1}-\binom{t_2-1}{1})=2t_1$.
$H_I(t_1,t_2)=\omega_1(t_1,t_2)+\omega_2(t_1,t_2)$=$2(t_1+1)+(t_2+1)-2$.
$A=\{(1,0),(0,1),(0,0)\}$. $\mu=(1,0)$, $\nu=(0,1)$, $a_{11}=0$, $a_{10}=2$ *and* $a_{01}=1$.

EXAMPLE 5: *Let $X=\{X_1,X_2\}$ and let $Y=\{Y_1\}$. Let σ_X and σ_Y be as in example 1 with $X_1>_{\sigma_X}X_2$. Let $I=(f_1=X_1^2-Y_1^2,f_2=X_2^2-Y_1^2)$. $\{f_1,f_2\}$ is L-autoreduced but it is not a characteristic set of I. $\{f_3=X_1^2-X_2^2,f_2=X_2^2-Y_1^2)$ is L-autoreduced and it is a characteristic set of I.*
$u_{f_3}=X_1^2$, $u_{f_2}=X_2^2$, $v_{f_3}=X_1^2$ and $v_{f_2}=Y_1^2$. $|U'_{t_1,t_2}|=4t_2+4$ and $|U''_{t_1,t_2}|=2\binom{t_1+2}{2}-4t_1-2$ when $t_1\geq 2$ and $t_2\geq 0$. So $\omega_1(t_1,t_2)=4t_2+4$ and $\omega_2(t_1,t_2)=2\binom{t_1+2}{2}-4t_1-2$.
$H_I(t_1,t_2)=\omega_1(t_1,t_2)+\omega_2(t_1,t_2)$= $2\binom{t_1+2}{2}+4(t_2+1)-4(t_1+1)+2$.
$A=\{(2,0),(1,0),(0,1),(0,0)\}$. $\mu=(2,0)$, $\nu=(0,1)$, $a_{21}=0$, $a_{20}=2$ *and* $a_{01}=4$.

References

BECKER, T., WEISPFENNING, V. (1993), *Gröbner Bases, A Computational Approach to Commutative Algebra*, Springer-Verlag GTM 141, (1993).

BUCHBERGER, B. (1976), 'A theoretical basis for the reduction of polynomials to canonical form', *ACM SIGSAM BULL.* **10** (3), 19-29.

BUCHBERGER, B., (1976), 'Some properties of Gröbner bases for polynomial ideals', *ACM SIGSAM BULL* **10**, 19-24.

CABOARA, M., DEDOMINICIS, G., ROBBIANO, L. (1996), 'Multigraded Hilbert Functions and Buchberger Algorithm', ACM Proceedings of ISSAC96, pp.72-85

CARRA' FERRO, G. (2001), 'Gröbner bases as Characteristic Sets', *Geometrical and Combinatorial Aspects of commutative Algebra* , *Lec. Notes in Pure and Applied Mathematics*, Marcel Dekker, Inc., pp.99-110.

KONDRATIEVA, M.V., LEVIN, A.B., MIKHALEV, A.V., PANKRATIEV, E.V. (1992), 'Computation of Dimension Polynomials' *Int. J. Algebra Comput.*, 117-137.

KONDRATIEVA, M.V., LEVIN, A.B., MIKHALEV, A.V., PANKRATIEV, E.V. (1999), *Differential and Difference Dimension Polynomials*, Kluwer Academic Publishers, Dordrecht (1999).

LEVIN, A.B. (1999), 'Computation of Hilbert Polynomials in Two Variables', *J. of Symb. Comp.* (28), 681-710.

RITT, J. F. (1950),*Differential Algebra*, AMS Coll.Publ. vol.33 New York (1950).

ROBBIANO, L., VALLA, G. (1998), 'Hilbert Series of Bigraded Algebras', Bollettino U.M.I. (8) 1-B, 521-540.

Using Gröbner bases in $\mathcal{D}$-modules theory*

Castro-Jiménez, F.J.[1] and Ucha, J.M.[1]

[1] *Depto. Álgebra, Universidad de Sevilla, Apdo. 1160, E-41080 Sevilla, Spain*

Abstract

Let D be a divisor in $\mathbf{C}^n$. We present computational methods to compare the $\mathcal{D}$-module of the meromorphic functions with respect to D to some natural approximations. By using the theory of Gröbner bases we show how the analytic case can be treated with computations in the Weyl algebra.

KEYWORDS: Gröbner basis, $\mathcal{D}$-module.

1. Introduction

Let us denote by $R_n = \mathbf{C}[x_1, \ldots, x_n]$ the complex polynomial ring in n variables and by A_n the Weyl algebra of order n. A_n is the associative $\mathbf{C}$-algebra generated by $2n$ symbols $x_1, \ldots, x_n, \partial_1, \ldots, \partial_n$ with relations

$$x_i x_j = x_j x_i,\ \partial_i \partial_j = \partial_j \partial_i,\ \partial_i x_j = x_j \partial_i + \delta_{ij}$$

where δ_{ij} is the Kronecker's symbol. A_n is isomorphic to the ring of linear differential operators over the ring R_n.

In the same way let us consider the ring $\mathcal{O}_n = \mathbf{C}\{x_1, \ldots, x_n\}$ of convergent power series in n variables and the ring $\mathcal{D}_n$ of linear differential operators with coefficients in $\mathcal{O}_n$. We have a natural inclusion $A_n \subset \mathcal{D}_n$.

An element P in A_n (resp. $\mathcal{D}_n$) is called a linear differential operator and it can be written as a finite sum

$$P = \sum_{\beta \in \mathbf{N}^n} p_\beta(x) \partial^\beta$$

where $\beta = (\beta_1, \ldots, \beta_n)$, $\partial^\beta = \partial_1^{\beta_1} \cdots \partial_n^{\beta_n}$ and $p_\beta(x) \in R_n$ (resp. $\mathcal{O}_n$).

If no confusion is possible we drop the index n and simply write R, A, $\mathcal{O}$ and $\mathcal{D}$. Remember that A (resp. $\mathcal{D}$) is a non-commutative, left and right noetherian

*Partially supported by BFM2001-3164 and FQM-813.

ring. Moreover, it is a simple ring (i.e. there are not non-trivial two-sided ideals in A_n), (see Björk (1979)).

In this paper we will study some A-modules (and some $\mathcal{D}$-modules) arising in a natural way from Algebraic Geometry. The ring R is a left A-module for the natural action defined as follows:

$$x_i \bullet f = x_i f,\ \partial_i \bullet f = \frac{\partial f}{\partial x_i}$$

for any $f \in R$. In fact, R is isomorphic, as an A-module, to the quotient of A by the left ideal generated by $\partial_1, \ldots, \partial_n$. In the same way $\mathcal{O}$ is a left $\mathcal{D}$-module.

Let us consider $f \in R$. The localization ring R_f (i.e. the ring of rational functions with poles along f) is the ring of quotients

$$R_f = \{\frac{g}{f^m} \,|\, g \in R,\ m \in \mathbf{N}\}.$$

R_f is a R-module and a left A-module in a natural way: the action $\partial_i \bullet \frac{g}{f^m}$ is just defined as the partial derivative of a rational function. Of course R_f is not a finitely generated R-module.

We have an analogous situation in the *analytic* setting, i.e. starting from $f \in \mathcal{O}$ and considering $\mathcal{O}_f$ (the ring of meromorphic functions with poles along f) as a left $\mathcal{D}$-module.

One of the main results in $\mathcal{D}$-module theory is the following theorem.

THEOREM 1.1: (Bernstein (1972), Björk (1979)) *For any $f \in R$ or $f \in \mathcal{O}$ we have:*
i) R_f is a finitely generated left A-module. In fact, there exists a positive integer number k such that R_f is the left A-module generated by the rational function $\frac{1}{f^k}$.
ii) $\mathcal{O}_f$ is a finitely generated left $\mathcal{D}$-module. In fact, there exists a positive integer number k such that $\mathcal{O}_f$ is the left $\mathcal{D}$-module generated by the meromorphic function $\frac{1}{f^k}$.

The left A-module generated by $\frac{1}{f^k}$ is just the set

$$A\frac{1}{f^k} = \{P \bullet \frac{1}{f^k},\ P \in A\} \subset R_f.$$

In computational $\mathcal{D}$-modules theory a natural problem is the following (for simplicity we only state the polynomial case):

Problem.- Given a polynomial $f \in R$:
a) Compute a positive integer number k such that $R_f = A\frac{1}{f^k}$ and
b) Compute a system of generators of the annihilator $Ann_A(1/f^k)$, i.e. compute a presentation

$$R_f = \frac{A}{Ann_A(\frac{1}{f^k})}.$$

The main ingredient in the proof of theorem 1.1 is the existence of the so called b-function (or Bernstein-Sato polynomial) attached to f (see Bernstein (1972), Björk (1979)), which is a non-zero polynomial $b_f(s) \in \mathbf{C}[s]$ with the following property: if $-k$ is the least integer root of $b_f(s)$ then

$$R_f = A\frac{1}{f^k}.$$

Bernstein proved (Bernstein (1972)) that the dimension of the characteristic variety of R_f is n, so R_f is *holonomic*. Kashiwara (Kashiwara (1978)) proved an analogous result for $\mathcal{O}_f$.

2. Gröbner bases in $\mathcal{D}$-modules theory

It will not be necessary to make a comprehensive development of the theory of Gröbner bases for $\mathcal{D}$-modules. In Briançon Maisonobe (1984) and Castro-Jiménez (1984, 1987) it is shown how the division and Buchberger's algorithm Buchberger (1965, 1970) can be adapted to the differential operators algebras A and $\mathcal{D}$. So the tools for computing Gröbner bases for left (and right) ideals and submodules of free modules, syzygies and free resolutions are available in this context. The book Saito et al. (2000) is an excellent introduction to Gröbner bases in A and its application to the study of GKZ-hypergeometric systems.

In $\mathcal{D}$ the situation is analogous but, as the coefficients of the differential operators can be convergent power series, the procedures are not algorithms in its precise sense.

The papers (Oaku Takayama (2001)), (Oaku et al. (2000)) contain deep applications of Gröbner bases to the effective computation of the four fundamental operations in $\mathcal{D}$–modules theory (localization, local cohomology, restriction and integration). These algorithms use as a main tool the effective computation of b-functions (Oaku (1997)).

The theory of Gröbner bases for A or $\mathcal{D}$ is part, in fact, of a more general theory of Gröbner bases in a certain family of non commutative rings, developed in Kandri-Rody Weispfenning (1990) (see also Bueso et al. (1998)).

3. Logarithmic vector fields. Meromorphic functions

We will enunciate some results in the context of $\mathcal{D}$-modules, but soon we will return to our effective calculations in A.

In this section we recall K. Saito's definition of logarithmic vector fields and we define some $\mathcal{D}$-modules–which are called logarithmic $\mathcal{D}$-modules– related to the $\mathcal{D}$-module of meromorphic functions $\mathcal{O}_f$.

For each point $p \in \mathbf{C}^n$ let us denote by $\mathcal{O}_p$ the ring of formal power series convergent in a neighborhood of p. Let us consider $Der(\mathcal{O}_p)$ the $\mathcal{O}_p$-module of $\mathbf{C}$-derivations of $\mathcal{O}_p$. The elements in $Der(\mathcal{O}_p)$ are called *vector fields*.

Let $D \subset \mathbf{C}^n$ be the divisor (i.e. the hypersurface) defined by a polynomial

$f \in R$ and $p \in D$. A vector field $\delta \in Der(\mathcal{O}_p)$ is said to be *logarithmic* with respect to D if $\delta(f) = af$ for some $a \in \mathcal{O}_p$. The $\mathcal{O}_p$-module of logarithmic vector fields (or logarithmic derivations) is denoted by $Der(\log D)_p$. If there exists a vector field δ such that $\delta(f) = f$ we will say that this divisor is *Euler homogeneous*. Quasi-homogeneous divisors (i.e. divisors defined by weighted homogeneous polynomials) are Euler homogeneous.

From now on, we will suppose that the origin $0 \in \mathbf{C}^n$ is in D and $p = 0$. We will consider some $\mathcal{D}$-modules associated to any divisor D (or more precisely to the germ $(D, 0)$). We call these modules logarithmic $\mathcal{D}$-modules.

- The (left) ideal $I^{\log D} \subset \mathcal{D}$ generated by the logarithmic vector fields $Der(\log D)$.
- The (left) ideal $\widetilde{I}^{\log D} \subset \mathcal{D}$ generated by the set $\{\delta + a \,|\, \delta \in I^{\log D} \text{ and } \delta(f) = af\}$. More generally, the ideals $\widetilde{I}^{(k)\log D}$ generated by the set

$$\{\delta + ka \,|\, \delta \in I^{\log D} \text{ and } \delta(f) = af\}$$

- The modules $M^{\log D} = \mathcal{D}/I^{\log D}$, $\widetilde{M}^{\log D} = \mathcal{D}/\widetilde{I}^{\log D}$ and more generally $\widetilde{M}^{(k)\log D} = \mathcal{D}/\widetilde{I}^{(k)\log D}$.

3.1. Logarithmic $\mathcal{D}$-modules and meromorphic functions

Logarithmic $\mathcal{D}$-modules are related to the $\mathcal{D}$-module of meromorphic functions $\mathcal{O}_f$ in the following way. The inclusion $\widetilde{I}^{(k)\log D} \subset Ann_{\mathcal{D}}(1/f^k)$ yields to a natural morphism $\phi_D^k : \widetilde{M}^{(k)\log D} \to \mathcal{O}_f$ defined by $\phi_D^k(\overline{P}) = P(1/f^k)$ where $\overline{P}$ denotes the class of the operator $P \in \mathcal{D}$ modulo $\widetilde{I}^{(k)\log D}$. The image of ϕ_D^k is $\mathcal{D}\frac{1}{f^k}$, i.e. the $\mathcal{D}$-submodule of $\mathcal{O}_f$ generated by $1/f^k$.

Considering the general ideals $\widetilde{I}^{(k)\log D}$ is a suggestion of Prof. Tajima. The point is the well known chain of inclusions

$$\mathcal{D}f^{-1} \subset \mathcal{D}f^{-2} \subset \cdots \subset \mathcal{D}f^{-k} = \mathcal{D}f^{-k-1} = \cdots = \mathcal{O}_f,$$

where $-k$ is least integer root of the b-function attached to f.

Under a computational point of view the divisor D will be defined by a polynomial $f \in R$ (more generally $D \subset \mathbf{C}^n$ could be an *analytic* divisor locally defined by germs of holomorphic functions, i.e. by convergent power series).

The b-function of the polynomial f is computable by Oaku's algorithm Oaku (1997) and there exists a direct method to give a presentation of the $\mathcal{D}$-module $\mathcal{O}_f$ Oaku Takayama (2001). If the calculation is intractable –as sometimes happens in the examples– we present here an indirect method to deduce that $\mathcal{O}_f$ and the modules $\widetilde{M}^{(k)\log D}$ do not coincide. The method is strongly based in the following result of Mebkhout (1989):

THEOREM 3.1: *The vector space $Ext^i_{\mathcal{D}}(\mathcal{O}_f, \mathcal{O})$ is zero for $i \geq 0$.*

More precisely, we prove that, under certain algorithmic conditions, some cohomology groups are not zero. It is the strategy used in Ucha (1999), Castro-Jiménez Ucha-Enríquez (2001) and Castro-Jiménez Ucha (2002).

Remark: It is important to underline that our methods manage the *analytic* case. As the inclusion $A \subset \mathcal{D}$ is flat (due to the flatness of the inclusion $R \subset \mathcal{O}$), the computation of syzygies and free resolutions in the Weyl algebra yields to the analogous computations in $\mathcal{D}$.

4. Comparison algorithms

We propose in this section two methods to compare the logarithmic modules presented above. It is important to remember that the computation of the *analytic* $Der(\log D)$ can be made for a divisor D if its equation is defined by a polynomial $f \in R$. Simply compute, using Gröbner basis, a system of generators of the module of syzygies $Syz(f_1, \ldots, f_n, f)$ where $f_i = \frac{\partial f}{\partial x_i}$ because the inclusion of the Weyl algebra in $\mathcal{D}$ is flat.

4.1. Direct comparison

The first method is complete but needs the calculation of the b-function.

Experimental evidences show that if the divisor is not *Euler homogeneous* (i.e. there is no $\delta \in Der(\log D)$ such that $\delta(f) = f$) the b-function is hard to compute. More precisely, the problem seems to be the calculation of the annihilator $Ann_{\mathcal{D}[s]}(f^s)$ and the use of certain elimination orders during the calculation of Gröbner bases (here $\mathcal{D}[s]$ stands for the polynomial ring, with the indeterminate s commuting with $\mathcal{D}$).

The following algorithm uses Oaku's algorithm Oaku (1997) for the computation of b-functions and Oaku-Takayama's algorithm computing annihilators Oaku Takayama (2001). The last two algorithms use different kinds of Gröbner basis computations in the Weyl algebra.

ALGORITHM 4.1: **INPUT:** *A polynomial equation $f = 0$ of a divisor $D \subset \mathbf{C}^n$;*

1. *Compute the b-function of f. Let $-\alpha_0$ be the least integer root.*
2. *Compute the ideal $Ann_{\mathcal{D}}(1/f^{\alpha_0})$.*
3. *Compute a set of generators $\{\mathbf{s}_1, \ldots, \mathbf{s}_r\}$ of $Syz(f_1, \ldots, f_n, f)$. The ideal $\widetilde{I}^{(\alpha_0)\log D}$ is generated by the elements*

$$\mathbf{s}_j \begin{pmatrix} \partial_1 \\ \vdots \\ \partial_n \\ -\alpha_0 \end{pmatrix} \in \mathcal{D}.$$

4. *Compare $Ann_{\mathcal{D}}(1/f^{\alpha_0})$ and $\widetilde{I}^{(\alpha_0)\log D}$.*

OUTPUT: $\mathcal{O}_f \simeq \widetilde{M}^{(\alpha_0)\log D} \Leftrightarrow Ann_{\mathcal{D}}(1/f^{\alpha_0}) = \widetilde{I}^{(\alpha_0)\log D}$.

The correctness of the algorithm is obvious as

$$\mathcal{O}_f \simeq \mathcal{D}\frac{1}{f^{\alpha_0}} \simeq \frac{\mathcal{D}}{Ann_{\mathcal{D}}(1/f^{\alpha_0})}.$$

4.2. Indirect approach: a sufficient condition

This second method is the alternative way when the computation of the b-function is intractable. Choose an integer $\alpha \geq 1$. To compare $\widetilde{M}^{(\alpha)\log D}$ with $\mathcal{O}_f$ we only need the computation of a free resolution of $\widetilde{M}^{(\alpha)\log D}$. As the algorithm looks for a technical condition in some step of the free resolution, in many examples it is not necessary to compute the whole resolution[†].

Definition: If

$$0 \to \mathcal{D}^{r_s} \xrightarrow{\varphi_s} \cdots \to \mathcal{D}^{r_2} \xrightarrow{\varphi_2} \mathcal{D}^{r_1} \xrightarrow{\varphi_1} \mathcal{D}^{r_0} \xrightarrow{\varphi_0} M \to 0$$

is a free resolution of a $\mathcal{D}$-module M, we say that the *Successive Matrices Condition (SMC) holds at level i* if the two successive morphisms φ_i, φ_{i+1} have matrices verifying:

1. There exists a column j in the matrix of φ_{i+1} whose elements are in the (left) ideal generated by $\partial_1, \ldots, \partial_n$.
2. The row j of the matrix of φ_i has all its entries of the form $\sum_\beta p_\beta(x)\partial^\beta$ with $p_\beta(0) = 0$ for any β. In particular, there are no "lonely constants" in the operators.

We will say that SMC holds (for the given free resolution) if it holds at some level i.

Algorithm 4.2: **INPUT:** *A polynomial equation $f = 0$ of a divisor $D \subset \mathbf{C}^n$;*

1. *Compute a set of generators $\{\mathbf{s}_1, \ldots, \mathbf{s}_r\}$ of $Syz(f_1, \ldots, f_n, f)$. For each positive integer α, the ideal $\widetilde{I}^{(\alpha)\log D}$ is generated by the elements*

$$\mathbf{s}_j \begin{pmatrix} \partial_1 \\ \vdots \\ \partial_n \\ -\alpha \end{pmatrix} \in \mathcal{D}.$$

2. *Compute a free resolution of $M = \widetilde{M}^{(\alpha)\log D}$*

$$0 \to \mathcal{D}^{r_s} \xrightarrow{\varphi_s} \cdots \to \mathcal{D}^{r_2} \xrightarrow{\varphi_2} \mathcal{D}^{r_1} \xrightarrow{\varphi_1} \mathcal{D} \xrightarrow{\pi} M \to 0.$$

[†]Taking into account that computing a complete free resolution can be a problem of great complexity, this option is very interesting.

OUTPUT: IF *SMC holds* **THEN** $\mathcal{O}_f \neq \widetilde{M^{(\alpha)\log D}}$.

We need a lemma to justify the algorithm. It explains the role of the SMC.

LEMMA 4.1: *Let D be a divisor, M a finitely generated left $\mathcal{D}$-module and*

$$0 \to \mathcal{D}^{r_s} \xrightarrow{\varphi_s} \cdots \to \mathcal{D}^{r_2} \xrightarrow{\varphi_2} \mathcal{D}^{r_1} \xrightarrow{\varphi_1} \mathcal{D}^{r_0} \xrightarrow{\pi} M \to 0 \quad (*)$$

a free resolution of M that satisfies SMC at level i. Then

$$Ext^i_{\mathcal{D}}(M, \mathcal{O}) \neq 0.$$

Proof: To obtain the Ext groups, we have to apply the functor $Hom_{\mathcal{D}}(-, \mathcal{O})$ to $(*)$. Using that

$$Hom_{\mathcal{D}}(\mathcal{D}^r, \mathcal{O}) \simeq \mathcal{O}^r$$

we obtain the complex

$$0 \to \mathcal{O}^{r_0} \xrightarrow{\varphi_1^t} \mathcal{O}^{r_1} \xrightarrow{\varphi_2^t} \mathcal{O}^{r_2} \to \cdots \xrightarrow{\varphi_{s-1}^t} \mathcal{O}^{r_{s-1}} \xrightarrow{\varphi_s^t} \mathcal{O}^{r_s} \to 0,$$

where φ_i^t denotes the morphism with matrix the transposed of φ_i. The derivatives now act naturally.

Then

$$Ext^i_{\mathcal{D}}(M, \mathcal{O}) = Ker\varphi_{i+1}^t / Im\varphi_i^t.$$

To finish the proof, the idea is obtaining an element in $Ker\varphi_{i+1}^t$ that is not in $Im\varphi_i^t$. This element yields a non zero element of $Ext^i_{\mathcal{D}}(M, \mathcal{O})$.

If the matrix of φ_{i+1} has no part in $\mathcal{O}$ in the j-th row, then $\mathbf{e} = (0, \ldots, 1, \ldots, 0)$ –where 1 is in the j-th position– is in $Ker\varphi_{i+1}^t$, as the derivatives applied to 1 are zero.

This element can not be in $Im\varphi_i^t$ if the matrix verifies the second condition in SMC. Applying the operators of the matrix it is not possible to obtain constants. □

We have the key to state the main result of this section: the correctness of the algorithm 4.2.

PROPOSITION 4.1: *Let $D \subset \mathbf{C}^n$ be a divisor with a free resolution of $\widetilde{M^{(\alpha)\log D}}$ that satisfies SMC at some level. Then $\mathcal{O}_f \neq \widetilde{M^{(\alpha)\log D}}$.*

Proof: Evident from lemma 4.1 (applied to $M = \widetilde{M^{(\alpha)\log D}}$) and theorem 3.1. □

A special case of SMC appears when you have a free resolution of length n (for a given $\mathcal{D}$-module M). That is a free resolution of type

$$0 \to \mathcal{D}^{r_n} \xrightarrow{\varphi_n} \cdots \to \mathcal{D}^{r_2} \xrightarrow{\varphi_2} \mathcal{D}^{r_1} \xrightarrow{\varphi_1} \mathcal{D}^{r_0} \xrightarrow{\pi} M \to 0$$

of length n. Then

$$Ext^n_{\mathcal{D}}(M, \mathcal{O}) = \frac{\mathcal{O}^{r_n}}{Im\varphi_n^t},$$

and SMC means, at level n, that there exists in the matrix of φ_n a row with no "lonely constants" (see 4.1).

Remark: Of course, natural generalizations of the SMC condition have to do with finding explicit elements in some $Ker\varphi_{i+1}^t$ with special properties and this seems to be difficult. Nevertheless, perhaps the results of Tsai Walther (2001) can be applied in this situation as follows:

$$Ext^0_A(R_f, R) \neq 0 \Rightarrow Ext^0_{\mathcal{D}}(\mathcal{O}_f, \mathcal{O}) \neq 0.$$

5. Application to the Spencer case.

In this section we explain how to apply the sufficient condition to a special case in which a specially tailored free resolution is provided.

Definition: (Saito (1980)) Let $D \subset \mathbf{C}^n$ be a divisor and suppose $0 \in D$. D is said to be *free* (at the origin) if the $\mathcal{O}$-module $Der(\log D)$ is free.

Smooth divisors and normal crossing divisors are free. By Saito (1980) any reduced germ of plane curve $D \subset \mathbf{C}^2$ is a free divisor. By Saito's criterium Saito (1980), $D \equiv (f = 0) \subset \mathbf{C}^n$ is free if and only if there exist n vector fields $\delta_i = \sum_{j=1}^n a_{ij}\partial_j$, $i = 1, \ldots, n$, such that $\det(a_{ij}) = uf$ where det means determinant and u is a unit in $\mathcal{O}$ (i.e $u(0) \neq 0$). Here ∂_j is the partial derivative $\frac{\partial}{\partial x_j}$ and a_{ij} is a holomorphic function in $\mathcal{O}$.

Definition: We say that a free divisor D is *of Spencer type* if the complex

$$\mathcal{D} \otimes_{\mathcal{O}} \wedge^{\bullet} Der(\log D) \to M^{\log D} \to 0$$

(introduced in Calderón-Moreno (1999)) is a (locally) free resolution of $M^{\log D}$ and if this last $\mathcal{D}$-module is holonomic.

There are analogous resolutions for the family of modules $\widetilde{M^{(k)\log D}}$.

For Spencer type divisors, the solution complex $Sol(M^{\log D})$ (that is, the complex $\mathbf{R}\mathcal{H}om_{\mathcal{D}}(M^{\log D}, \mathcal{O})$) is naturally quasi-isomorphic to $\Omega^{\bullet}(\log D)$ (as we pointed in Castro-Jiménez Ucha (2002) as a deduction of Calderón-Moreno (1999)). Here $\Omega^{\bullet}(\log D)$ is the complex of logarithmic differential forms with respect to D (see Saito (1980)). On the other hand, the duality (in the sense of $\mathcal{D}$-modules) $(M^{\log D})^* \simeq \widetilde{M^{\log D}}$ proved in Castro-Jiménez Ucha (2002) has important consequences comparing $\widetilde{M^{\log D}}$ and $\mathcal{O}_f$, namely:

THEOREM 5.1: (Ucha (1999), Castro-Jiménez Ucha-Enríquez (2001)) *In dimension 2, the morphism $\phi_D^1 : \widetilde{M}^{\log D} \to \mathcal{O}_f$ (see 3.1) is an isomorphism if and only if $D \equiv (f = 0)$ is a quasi-homogeneous plane curve.*

THEOREM 5.2: (Castro-Jiménez Ucha (2002)) *Suppose the divisor $D \subset \mathbf{C}^n$ is free and locally quasi-homogeneous. Then the morphism $\phi_D^1 : \widetilde{M}^{\log D} \to \mathcal{O}_f$ (see 3.1) is an isomorphism (so, $\widetilde{M}^{\log D}$ and $\mathcal{O}_f$ are isomorphic as $\mathcal{D}$-modules).*

The methods presented in section 4 give us computational tools to check the comparison between $\widetilde{M}^{\log D}$ and $\mathcal{O}_f$.

Remark: Once you have the duality of Castro-Jiménez Ucha (2002), you also have a strategy to test if the so called *Logarithmic Comparison Theorem (LCT)* holds, that is, if the complex $\Omega^\bullet(\star D)$ of meromorphic differential forms and the complex $\Omega^\bullet(\log D)$ of logarithmic differential forms (both with respect to D) are quasi isomorphic (see Calderón Moreno et al. (2002) and Castro-Jiménez et al. (1996)).

Remark: There are two interesting experimental suggestions:

- We don't know examples of free divisors with integer roots of their b-function less than -1.
- We only know free divisors of Spencer type.

6. Examples

It is very important to point out that all the calculations needed in this section are *calculations of Gröbner Bases*, namely

- Computations of syzygies among a polynomial and its derivatives to present $I^{\log D}$ or $\widetilde{I}^{\log D}$.
- Test if a divisor is Euler homogeneous: the property holds if the ideal of first components of elements in $Syz(f, \frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n})$ is the whole ring.
- Computations of free resolutions (so syzygies again, essentially) of modules over the correspondent Weyl algebra.
- Equality of left ideals in A.
- Calculation of $Ann_{\mathcal{D}}(1/f)$ and b-function for a polynomial f: Gröbner bases with elimination orders in the corresponding Weyl algebra with an additional variable s (see Saito et al. (2000) for example).
- Test holonomicity of a $\mathcal{D}$-module, i.e. test if the attached characteristic variety has dimension n (see Saito et al. (2000) too).

All over the section, the computations have been made with `kan/sm1` and the $\mathcal{D}$-modules package of `Macaulay 2` (respectively Takayama (1991) and Grayson Stillman (1999)). Finally, some computations of syzygies among polynomials have been made with `CoCoA` (see Capani et al. (1995)).

6.1. Example 1: $D \equiv (x(x^2-y^3)(x^2-zy^3)=0) \subset \mathbf{C}^3$

We treat here the divisor $D \subset \mathbf{C}^3$ whose local equation at $(0,0,0)$ is given by $f=0$ with

$$f = x(x^2-y^3)(x^2-zy^3).$$

This example belongs to an interesting family: it is not locally quasi-homogeneous[‡] but *Euler homogeneous* and verifies that $Ann_{\mathcal{D}}(1/f) = \widetilde{I}^{\log D}$. Remember that in dimension 2 the last equality and to be quasi-homogeneous are equivalent.

This divisor is free and $\delta_1, \delta_2, \delta_3$ form a (global) basis of $Der(logD)$, where

$$\begin{array}{rcl} \delta_1 & = & \frac{3}{2}x\partial_x + y\partial_y \\ \delta_2 & = & (y^3z - x^2)\partial_z \\ \delta_3 & = & (-\frac{1}{2}xy^2)\partial_x - \frac{1}{3}x^2\partial_y + (y^2z^2 - y^2z)\partial_z, \end{array}$$

whose coefficients verify that

$$\begin{vmatrix} \frac{3}{2}x & y & 0 \\ 0 & 0 & y^3z-x^2 \\ -\frac{1}{2}xy^2 & -\frac{1}{3}x^2 & y^2z^2-y^2z \end{vmatrix} = -\frac{1}{2}f.$$

This example illustrates the direct approach: we can calculate the annihilator of $1/f$ –because it is manageable– and compare it with $\widetilde{I}^{\log D}$. They are the same ideal. We calculate the b-function of f too. Its least integer root is -1, so

$$\mathcal{O}_f \simeq \mathcal{D}\cdot f^{-1} \simeq \frac{\mathcal{D}}{Ann_{\mathcal{D}}(1/f)}.$$

To complete this example, we will explain how the duality and the Spencer type condition mentioned in 5 are checked. We use for this purpose a free resolution of $M^{\log D}$.

Our work is divided in two steps:

- Verify that $M^{\log D}$ is holonomic: if it is not holonomic, the computation of its dual can not be managed as we do. In this case the dimension of the characteristic variety of $M^{\log D}$ is 3, so it is holonomic.
- Compute a free resolution of $M^{\log D}$ and check if it is of Spencer type. If this happens then duality holds by Castro-Jiménez Ucha (2002).

Here are some details of the resolution:

1. The module $Syz(\delta_1,\delta_2,\delta_3)$ is generated by the syzygies obtained from the commutators $[\delta_i,\delta_j]$. We have $Syz(\delta_1,\delta_2,\delta_3) = \langle \mathbf{s}_{12}, \mathbf{s}_{13}, \mathbf{s}_{23}\rangle$ where

$$\begin{array}{rcl} \mathbf{s}_{12} & = & (-\delta_2, \delta_1 - 3, 0) \\ \mathbf{s}_{13} & = & (-\delta_3, 0, \delta_1 - 2) \\ \mathbf{s}_{12} & = & (0, -\delta_3 - y^2z, \delta_2). \end{array}$$

[‡]Out of the scope of this work, there is an indirect proof of this fact using that it is not a *Koszul free* divisor (see Calderón-Moreno Narváez-Macarro (1999)).

2. The module $Syz(\mathbf{s}_{12}, \mathbf{s}_{13}, \mathbf{s}_{23})$ is generated by the element $\mathbf{r}$:

$$\mathbf{r} = (-y^2z^2\partial_z + y^2z\partial_z + \frac{1}{2}xy^2\partial_x - y^2z + \frac{1}{3}x^2\partial_y,\ \ y^3z\partial_z - x^2\partial_z,\ \ -y\partial_y - \frac{3}{2}x\partial_x + 5)$$

This is the element required to have the Spencer type resolution so duality holds.

6.2. Example 2: $D \equiv ((xz+y)(x^4+y^5+xy^4) = 0) \subset \mathbf{C}^3$

In this example the divisor $D \subset \mathbf{C}^3$ has as a local equation at $(0,0,0)$ the form

$$f = (xz+y)(x^4+y^5+xy^4) = 0.$$

The divisor is free with $\delta_1, \delta_2, \delta_3$ as a global basis of $Der(logD)$:

$$\begin{aligned}
\delta_1 &= xz\partial_z + y\partial_z + x \\
\delta_2 &= -8x^2\partial_x - 10xy\partial_x - 6xy\partial_y - 8y^2\partial_y + 2yz\partial_z - 2y\partial_z - 40x - 48y \\
\delta_3 &= -1/4y^2z^2\partial_z - xy^2\partial_x - 1/4y^3\partial_x - 3/4y^3\partial_y + 1/4y^2z\partial_z - 1/4y^2z - \\
&\quad -5/4x^2\partial_x + 25/4xy\partial_x + 1/4x^2\partial_y - 5/4xy\partial_y + 5y^2\partial_y - 5/4yz\partial_z - 19/4y^2 - \\
&\quad -1/4x\partial_z - 25/4x + 30y
\end{aligned}$$

In this case, it can be checked that D is of Spencer type and the third matrix (corresponding to the second module of syzygies) is (P_1, P_2, P_3) where

$$\begin{aligned}
P_1 &= -2y^2z + 2y^2 + 2x + 20y \\
P_2 &= y^2z^2\partial_z + 4xy^2\partial_x + y^3\partial_x + 3y^3\partial_y - y^2z\partial_z + y^2z + 5x^2\partial_x - 25xy\partial_x - x^2\partial_y + \\
&\quad +5xy\partial_y - 20y^2\partial_y + 5yz\partial_z + 16y^2 + x\partial_z + 20x - 85y \\
P_3 &= -32x^2\partial_x - 40xy\partial_x - 24xy\partial_y - 32y^2\partial_y + 8yz\partial_z - 8y\partial_z - 120x - 136y
\end{aligned}$$

It has no constants thus, again we have that $Ext^3_{\mathcal{D}}(\widetilde{M}^{\log D}, \mathcal{O}) \neq 0$ because SMC holds at level 3. So, $\mathcal{O}_f$ and $\widetilde{M}^{\log D}$ do not coincide.

Remark: It seems that if D is a free and not Euler homogeneous divisor then $Ann_{\mathcal{D}}(1/f)$ and $\widetilde{I}^{\log D}$ do not coincide. For this family the SMC holds at last level in *all* the examples we have tried. In fact, in dimension 2 the proof of 5.1 uses the SMC to compare the logarithmic ideals and the annihilators. We are working on a generalization of this result which would have important consequences in the context of the logarithmic comparison theorem.

6.3. Example 3: $D \equiv (xyz(x+y)(x+z)=0) \subset \mathbf{C}^3$

This example is an application of theorem 5.2. The direct calculation of $Ann_{\mathcal{D}}(1/f)$ –where f is the local equation of the arrangement of (hyper)planes– is not manageable in the versions of computers systems for $\mathcal{D}$-modules available for us. Our approach is then very helpful.

A very interesting study of the b-functions of these arrangements is in Walther (2002). Precisely, we think that theorem 5.2 will be a short cut to prove the last conjecture of Walther (2002) in the free case.

We have to check:

- The divisor is certainly free: a (global) basis of $Der(\log D)$ is

$$\begin{aligned}\delta_1 &= 5 + x\partial_x + y\partial_y + z\partial_z \\ \delta_2 &= 12x + 9z + (3x^2 + 3xz)\partial_x + (3xy + 3yz)\partial_y \\ \delta_3 &= -9/2x + 3y - 9/2z + (-3/2x^2 - 3/2xz)\partial_x + (3/2y^2 - 3/2yz)\partial_y,\end{aligned}$$

  whose associated determinant of coefficients is equal to $-9/2 \cdot f$.
- The b-function is not manageable as well. Fortunately we have a theorem of Leykin (see Walther (2002)) that assures that the least integer root of the b-function of an arrangement is -1. So $\mathcal{O}_f \simeq \mathcal{D} \cdot f^{-1}$.

The last two conditions lead us to the presentation of $\mathcal{O}_f$ using that $Ann_{\mathcal{D}}(1/f) = \widetilde{I}^{\log D}$.

6.4. Example 4: $D \equiv (f=0) \subset \mathbf{C}^n$ **with** $f = x_1^n + x_2^n + \cdots + x_n^n$**.**

In this example is well known that the least integer root of f is $-n+1$ so in this case

$$\mathcal{O}_f = \mathcal{D} \cdot \frac{1}{f^{n-1}}.$$

On the other hand, it is not a free divisor for $n \geq 2$ (because D has an isolated singularity) so we do not have theorems to apply! For this example is very easy to apply the direct approach. However, we will use this example to enlighten in a concrete case the Successive Matrices Condition too.

The direct method in this case works like this:

• The ideal $Ann_{\mathcal{D}[s]}(f^s)$ is generated by

$$\{-ns + x_1\partial_1 + \cdots + x_n\partial_n,\ \frac{\partial f}{\partial x_i}\partial_j - \frac{\partial f}{\partial x_j}\partial_i \text{ for } 1 \leq i < j \leq n\},$$

as you can easily check using the algorithm of Oaku[§] (see Oaku (1997)). Specializing s to the value $s = -n+1$ leads us to deduce that

$$Ann(1/f^{n-1}) = \langle -n(-n+1) + x_1\partial_1 + \cdots + x_n\partial_n \rangle + \langle \frac{\partial f}{\partial x_i}\partial_j - \frac{\partial f}{\partial x_j}\partial_i \text{ for } 1 \leq i < j \leq n \rangle.$$

[§]The calculation of $Ann_{\mathcal{D}[s]}(f^s)$ and the b-function can be made by hand with Gröbner bases for the general case.

As a set of generators of $Syz(\frac{\partial f}{\partial x_1},\ldots,\frac{\partial f}{\partial x_n},f)$ is

$$\{(x_1,\ldots,x_n,-n)\}\bigcup\{\frac{\partial f}{\partial x_i}\mathbf{e}_{j+1}-\frac{\partial f}{\partial x_j}\mathbf{e}_{i+1}\ \text{ for } 1\leq i<j\leq n\},$$

(where $\mathbf{e}_l$ is the element of $A_n(\mathbf{C})^{n+1}$ with 1 in the position l and 0 in the rest, for $1\geq l\geq n$).

We have that

$$\mathcal{O}_f\simeq\mathcal{D}\frac{1}{f^{n-1}}=\widetilde{M}^{(n-1)\log D},$$

as it can be deduced comparing the annihilator $Ann_{\mathcal{D}}(1/f^{n-1})$ with $\widetilde{I}^{(n-1)\log D}$. They are the same ideal.

• We will use this example to illustrate the SMC for $n=3$ (i.e. for $f=x^3+y^3+z^3$) at levels 2 and 3 in order to compare $Ann_{\mathcal{D}}(1/f)$ with $\widetilde{I}^{\log D}$.

Remark: : It is remarkable that sometimes the SMC is hidden. If you apply the command `Dres` of the $\mathcal{D}$-modules package of `Macaulay 2` the resolution obtained does not verify the SMC. Instead of the resolution provided directly, the use of the command `kernel` to control exactly which are the generators chosen in each step, is very friendly to look for the needed conditions.

In this case the following free resolution of $\widetilde{M}^{\log D}$ can be obtained:

$$0\to\mathcal{D}^3\xrightarrow{\varphi_2}\mathcal{D}^6\xrightarrow{\varphi_1}\mathcal{D}^4\xrightarrow{\varphi_0}\mathcal{D}\xrightarrow{\pi}\widetilde{M}^{\log D}\to 0$$

The matrix φ_1 of the first module of syzygies is :

$$\begin{pmatrix} 0 & \partial_x & -\partial_y-\partial_z & \\ 0 & x^2 & -y^2 & z^2 \\ z^2\partial_y-y^2\partial_z & -3y\partial_y-3z\partial_z-6 & -3x\partial_y & x\partial_z \\ z^2\partial_x-x^2\partial_z & 0 & -3x\partial_x-3y\partial_y-3z\partial_z-6 & 0 \\ y^2\partial_x-x^2\partial_y & 0 & 0 & -3x\partial_x-3y\partial_y-3z\partial_z-6 \\ 0 & -2x & y^2\partial_x-x^2\partial_y & -z^2\partial_x+x^2\partial_z \end{pmatrix}.$$

The matrix of the second module of syzygies is

$$\varphi_2=\begin{pmatrix} -x^2 & \partial_x & 0 & 0 & 0 & 1 \\ -3y\partial_y-3z\partial_z-6 & 0 & -\partial_x & \partial_y & -\partial_z & 0 \\ 0 & 3y\partial_y+3z\partial_z & x^2 & -y^2 & z^2 & -3x \end{pmatrix}.$$

As you can easily detect, the second row of φ_1 is

$$(0,x^2,-y^2,z^2),$$

with no constants. In φ_2 the corresponding second column is

$$\begin{pmatrix} \partial_x \\ 0 \\ 3y\partial_y+3z\partial_z \end{pmatrix},$$

with all its elements in the ideal generated by $\partial_x, \partial_y, \partial_z$. The SMC holds at level 2 so $Ext^2_{\mathcal{D}}(\widetilde{M}^{\log D}, \mathcal{O}) \neq 0$.

As the third row of φ_2 is

$$(0, 3y\partial_y + 3z\partial_z, x^2, -y^2, z^2, -3x)$$

and there are no constants, $Ext^3_{\mathcal{D}}(\widetilde{M}^{\log D}, \mathcal{O}) \neq 0$ too. It is the SMC at level 3.

7. Acknowledgements

We thank Prof. Tajima and Prof. David Mond for very helpful ideas and comments.

References

Bernstein, I. N. (1972), 'Analytic continuation of generalized functions with respect to a parameter', *Funkcional. Anal. i Priložen.* **6**(4), 26–40.

Björk, J.-E. (1979), *Rings of differential operators*, Vol. 21 of *North-Holland Mathematical Library*, North-Holland Publishing Co., Amsterdam.

Briançon, J. Maisonobe, P. (1984), 'Idéaux de germes d'opérateurs différentiels à une variable', *L'Enseignement Math.* **30**, 7–38.

Buchberger, B. (1965), An Algorithm for Finding the Bases Elements of the Residue Class Ring Modulo a Zero Dimensional Polynomial Ideal (German), PhD thesis, Univ. of Innsbruck (Austria).

Buchberger, B. (1970), 'An Algorithmical Criterion for the Solvability of Algebraic Systems of Equations (German)', *Aequationes Mathematicae* **4**(3), 374–383.

Bueso, J. L., Gómez Torrecillas, J., Lobillo, F. J. Castro, F. J. (1998), An introduction to effective calculus in quantum groups, *in* 'Rings, Hopf algebras, and Brauer groups (Antwerp/Brussels, 1996)', Vol. 197 of *Lecture Notes in Pure and Appl. Math.*, Dekker, New York, pp. 55–83.

Calderón-Moreno, F. (1999), 'Logarithmic differential operators and logarithmic de Rham complexes relative to a free divisor', *Ann. Sci. École Norm. Sup. (4)* **32**(5), 701–714.

Calderón Moreno, F. J., Mond, D., Narváez Macarro, L. Castro Jiménez, F. J. (2002), 'Logarithmic cohomology of the complement of a plane curve', *Comment. Math. Helv.* **77**(1), 24–38.

Calderón-Moreno, F. Narváez-Macarro, L. (1999), Locally quasi-homogeneous free divisors are koszul free, Technical Report 56, Prepub. de la Fac. de Matemáticas. Univ. de Sevilla.

Capani, A., Niesi, G. Robbiano, L. (1995), 'CoCoa, a system for doing computations in Commutative Algebra. Available via anonymous ftp from: `cocoa.dima.unige.it`'.

Castro-Jiménez, F. (1984), Théorème de division pour les opérateurs différentiels et calcul des multiplicités, Thèse de 3^{eme} cycle, Univ. of Paris VII.

Castro-Jiménez, F. (1987), Calculs effectifs pour les idéaux d'opérateurs différentiels, *in* 'Actas de la II Conferencia Internacional de Geometría Algebraica. La Rábida.', Travaux en Cours 24, Hermann.

Castro-Jiménez, F. J., Narváez-Macarro, L. Mond, D. (1996), 'Cohomology of the complement of a free divisor', *Trans. Amer. Math. Soc.* **348**(8), 3037–3049.

Castro-Jiménez, F. J. Ucha-Enríquez, J. M. (2001), 'Explicit comparison theorems for D-modules', *J. Symbolic Comput.* **32**(6), 677–685. Effective methods in rings of differential operators.

Castro-Jiménez, F. J. Ucha, J. M. (2002), 'Free divisors and duality for D-modules', *Proc. Steklov Inst. Math.* **238**, 88–96.

Grayson, D. Stillman, M. (1999), 'Macaulay 2: a computer algebra system for algebraic geometry, version 0.9.2, `http://www.math.uiuc.edu/macaulay2`. D-module scripts by A. Leykin and H. Tsai, `http://www.math.cornell.edu/ htsai/`'.

Kandri-Rody, A. Weispfenning, V. (1990), 'Noncommutative Gröbner bases in algebras of solvable type', *J. Symbolic Comput.* **9**(1), 1–26.

Kashiwara, M. (1978), 'On the holonomic systems of linear differential equations. II', *Invent. Math.* **49**(2), 121–135.

Mebkhout, Z. (1989), *Le formalisme des six opérations de Grothendieck pour les D-modules cohérents*, Vol. 35 of *Travaux en Cours*, Hermann, Paris.

Oaku, T. (1997), 'An algorithm of computing b-functions', *Duke Math. J.* **87**(1), 115–132.

Oaku, T. Takayama, N. (2001), 'Algorithms for D-modules—restriction, tensor product, localization, and local cohomology groups', *J. Pure Appl. Algebra* **156**(2-3), 267–308.

Oaku, T., Takayama, N. Walther, U. (2000), 'A localization algorithm for D-modules', *J. Symbolic Comput.* **29**(4-5), 721–728. Symbolic computation in algebra, analysis, and geometry (Berkeley, CA, 1998).

Saito, K. (1980), 'Theory of logarithmic differential forms and logarithmic vector fields', *J. Fac. Sci. Univ. Tokyo Sect. IA Math.* **27**(2), 265–291.

Saito, M., Sturmfels, B. Takayama, N. (2000), *Gröbner deformations of hypergeometric differential equations*, Vol. 6 of *Algorithms and Computation in Mathematics*, Springer-Verlag, Berlin.

Takayama, N. (1991), '**Kan:** a system for computation in algebraic analysis. source code available for unix computers from `ftp.math.kobe-u.ac.jp`'.

Tsai, H. Walther, U. (2001), 'Computing homomorphisms between holonomic D-modules', *J. Symbolic Comput.* **32**(6), 597–617. Effective methods in rings of differential operators.

Ucha, J. (1999), Métodos constructivos en álgebras de operadores diferenciales., PhD thesis, Univ. of Sevilla.

Walther, U. (2002), Bernstein-Sato polynomial versus cohomology of the Milnor fiber for generic arrangements, Technical report, Preprint arXiv:math.AG/0204080.

Minimal generators from reduced Gröbner bases obtained by interpolation methods

Francesca Cioffi[1] and Ferruccio Orecchia[1]

[1] *Dip. di Matematica e Applicazioni "R. Caccioppoli", Università di Napoli "Federico II", Via Cintia, 80126 Napoli - Italy*

Abstract

In 1982 Buchberger and Möller described a polynomial algorithm to compute a reduced Gröbner basis of the ideal of affine points relying on interpolation methods. This algorithm was an incisive step for a worthwhile progress in computation of zero-dimensional schemes and their applications. The consequent generalization of the original algorithm of Buchberger and Möller to projective points gave space to the problem of minimalizing a homogeneous Gröbner basis even without computing syzygies. Answers to this problem have been already given and applied in several particular cases. The purpose of this paper is to illustrate in a more general context and to improve a method to minimalize reduced Gröbner bases that we have already successfully used in several applications of zero-dimensional computation.

KEYWORDS: minimal generators, Gröbner bases, Buchberger-Möller algorithm

1. Introduction

The Gröbner basis of a polynomial ideal I is a particular set of generators of I introduced by Buchberger in 1965 together with an algorithm for computing it (see, for example, Buchberger (1998) and references therein for a survey about Gröbner bases). It is well known that this algorithm takes any set of generators of I as input and is based on a characterization of a Gröbner basis by S-polynomials. This means that the construction of the Gröbner basis involves the construction of syzygies in such a way that minimal generators of a homogeneous ideal I can be singled out among the polynomials of the Gröbner basis itself. Hence, in this case the minimalization of a Gröbner basis of a homogeneous ideal I can occur while the Gröbner basis is constructed.

However, the Gröbner basis can be also constructed by the algorithm of Buchberger and Möller (1982) which takes a set of points as input and is based on

interpolation methods. In this case it makes sense to ask for a minimalization of a reduced Gröbner basis. Answers to this problem have been given in (Marinari et al. 1993, Cioffi 1996, 1999). Although they are described in the particular case of computation on points, they work in a more general context. As the study of zero-dimensional schemes can give information also on positive dimensional schemes, in several cases the algorithm of Buchberger and Möller, which is applied to points in (Marinari et al. 1993, Abbott et al. 2000, 2001, Cioffi Orecchia 2001), turns out to be useful also for positive dimensional varieties. For example, it gave rise to implicitization methods that are alternative to the Gröbner bases technique (Albano et al. 2000, Orecchia 2001*b*), to an algorithm that constructs elliptic curves (Chiantini et al. 2001) and it is a tool for the study of minimally generated rational curves (Orecchia 2001*a*).

Moreover, the fact that in (Berry 1997, 2000) the shape of reduced Gröbner bases of some affine space curves is deduced by geometric tools shows that in some cases a Gröbner basis can be constructed by alternative methods.

For these reasons, in this paper we highlight the applicability of the method of minimalization arisen for points in (Cioffi 1996, 1999) to any homogeneous ideal. Moreover, we introduce an improvement, described in detail in section 4. An implementation of this algorithm, prepared in the object-oriented language C++ over a field K of positive characteristic p and compiled using g++ and NTL 5.2 (Shoup 2001), is available at `http://cds.unina.it/~cioffifr`. It is applicable to reduced Gröbner bases with respect to (w.r.t.) the graded reverse lexicographic (deg-rev-lex) term order, because this order is the best suited for minimalization (Bayer Stillman 1987). In section 3 we illustrate the algorithm and in section 5 we report some tests.

2. Notation and basic facts

Let $S = K[x_0, \ldots, x_r]$ be the ring of polynomials in $r+1$ variables over a field K, S_d be the K-vector space that consists of the homogeneous polynomial of S of degree d and $I = \bigoplus_{d\geq 0} I_d$ be a homogeneous ideal of S, where I_d is the K-vector space of all homogeneous polynomials of degree d of I. Then $S/I = \bigoplus_{d\geq 0}(S/I)_d \tilde{=} \bigoplus_{d\geq 0} S_d/I_d$ is a graded algebra and its Hilbert function is the numerical function $H_{S/I} : \mathbb{N} \to \mathbb{N}$ such that $H_{S/I}(d) := dim_K(S_d/I_d)$.

Recall that the number of minimal homogeneous generators of I is an invariant for I. Indeed the number of minimal homogeneous generators of I of a given degree d is one of the invariants called graded Betti numbers of I, and it appears in the first step of the minimal free resolution of I. Since one fundamental step in the construction of the minimal free resolution of I is just the computation of minimal generators of submodules of a free module over S, we point out that the described algorithm is applicable also to such submodules.

In (Ramella 1990) a method is described to compute minimal generators of an ideal I of projective points without using the theory of Gröbner bases: if B_d is a K-basis of the K-vector space I_d, then the dependent generators belonging to B_d

are detected by checking linear dependences in the set $\{x_0, \ldots, x_r\} \cdot B_{d-1} \cup B_d$. This method has a computational cost that is polynomial in the number of points and in the number $r+1$ of variables if the points are in generic position. And the minimalization step is applicable to any ideal with the condition that it is possible to construct a basis B_d.

As we reminded in the introduction, in the construction of Gröbner bases by the algorithm of Buchberger it is classical to get minimal generators by looking at the constant components of the syzygies: if $\{f_1, \ldots, f_t\}$ is a set of homogeneous generators of I and $Syz(I) = \{(h_1, \ldots, h_t) \in S^t | \sum_{i=1}^{t} h_i f_i = 0\}$ is a first module of syzygies of I, then a polynomial f_k depends if and only if there is a homogeneous generator $(h_1, \ldots, h_t)$ of $Syz(I)$ such that $h_k \in K$. There are many improvements of this approach and one can see Caboara et al. (2002) for a very recent result. However, in this paper we are interested in minimalizing Gröbner bases computed by the Buchberger and Möller algorithm.

In (Marinari et al. 1993) an algorithm is described that minimalizes a reduced Gröbner basis of a homogeneous ideal by computing only the constant components of the syzygies. Its computational cost is evaluated for ideals defined by functionals and is polynomial in the number of functionals and in the number $r+1$ of variables.

The algorithm that now we are going to illustrate here does not compute syzygies or a piece of them, although it exploits a result of Marinari et al. (1993) about syzygies for obtaining that in the algorithm of Ramella (1990) it is sufficient to consider only a subset of particular bases B_{d-1} and B_d.

Recall that I is a homogeneous ideal of the polynomial ring $S = K[x_0, \ldots, x_r]$ in $r+1$ variables over a field K and let $\mathcal{T} = \{x_0^{\alpha_0} \ldots x_r^{\alpha_r} \mid (\alpha_0, \ldots, \alpha_r) \in \mathbb{N}^{r+1}\}$ be the set of all terms of S ordered with respect to a term order $<$, i.e. a monoid order on $\mathcal{T}$ for which 1 is the smallest term. We say that a term T is a *multiple* of a term H if there is a term $\bar{H}$ such that $T = H\bar{H}$. If $\bar{H}$ is a variable, H is a *predecessor* of T. If p is a polynomial of $S = K[x_0, \ldots, x_r]$, $LT(p)$ is the leading term of p. If $\mathcal{P}$ is a set of polynomials of S, $LT(\mathcal{P})$ is the ideal generated in S by the leading terms of the polynomials of $\mathcal{P}$, and $\mathcal{P}_d$ is the set of homogeneous elements of $\mathcal{P}$ of degree d.

Recall that a set $G \subset I$ of non-null polynomials of I is a Gröbner basis of I, with respect to a given term order $<$, if $LT(G) = LT(I)$. From the definition it follows that a Gröbner basis is a set of generators. A Gröbner basis is called *reduced* if its polynomials are monic and any term with a non-null coefficient in one of the polynomials of G is not divided by a leading term of a polynomial of G.

A rewriting procedure, with respect to a finite subset F of S and to a given term order, is an algorithmic method for substituting a polynomial f with a polynomial f' (called *remainder*) such that $f - f'$ belongs to the ideal generated by F and such that every coefficient of a term that belongs to $LT(F)$ vanishes in f'. The polynomial f' is unique if F is a Gröbner basis (see the Division Algorithm, for example, in (Kreuzer Robbiano 2000) Th. 1.6.4).

If G is a Gröbner basis of I and T is a term which belongs to the monomial ideal $LT(G)$, then the *normal form* $NF(T)$ of T is the remainder of the Division Algorithm of T with respect to G. Hence, it is a linear combination of terms not belonging to $LT(G)$ such that $b(T) := T - NF(T)$ belongs to I. If a term T belongs to the ideal $LT(G)$, then we say that T is *dependent*, otherwise we say that it is *independent*.

From now we fix only graded term orders, i.e. term orders by which terms are compared first w.r.t. their total degrees. Let $B_d = \{b(T)|T \in LT(G)_d\}$ be the set of the polynomials of type $b(T)$ where T is a dependent term of degree d. If $G = \{f_1, \ldots, f_t\}$ is the reduced (homogeneous) Gröbner basis of I with respect to a graded term order $<$, then $G_d \subset B_d$.

Remark: The set $B_d = \{b(T)|T \in LT(G)_d\}$ is a K-basis of I_d. In fact, the definition of Gröbner basis itself makes possible to rewrite any polynomial of I_d by polynomials of B_d.

From now we suppose that G is reduced and that a and b are respectively the minimum and the maximum degree of the polynomials of G.

Let $\bar{B}_d = \{b(T) \in B_d \mid \exists x_i : \frac{T}{x_i} \notin LT(G)\} \subset B_d$ be the set of the polynomials of B_d whose leading term has at least one independent predecessor. Then

$$G_d = \{b(T) \in \bar{B}_d \mid \frac{T}{x_i} \notin LT(G),\ \forall i = 0, \ldots, r\}.$$

Remark: Since $H_{S/I}(d) = H_{S/LT(I)}(d)$ (Macaulay 1927) it follows that

$$H_{S/I}(d) = |\{T \in \mathcal{T}_d \mid T \notin LT(G)\}|$$

and, hence, $|\bar{B}_d| \leq r \cdot |\{T \in \mathcal{T}_{d-1} \mid T \notin LT(G)\}| = r \cdot H_{S/I}(d-1)$.

3. Minimalization without computing syzygies

With the notation already introduced in section 2, let

$$T_i = LT(f_i), \quad T_{i,j} = l.c.m.(LT(f_i), LT(f_j)),$$

$$f_{i,j} = (T_{i,j}/T_i)\ f_i - (T_{i,j}/T_j)\ f_j.$$

It is well known that, from all the identities of type $f_{i,j} = \sum_{k=1}^{t} h_k f_k$, where $LT(h_k f_k) \leq LT(f_{i,j}) < T_{i,j}$, it is possible to obtain the following set of generators of the syzygies of G (see, for example, Kreuzer Robbiano (2000)):

$$\{(-h_1, \ldots, -h_{i-1}, \frac{T_{i,j}}{T_i} - h_i, \ldots, -h_{j-1}, \frac{T_{i,j}}{T_j} - h_j, \ldots, -h_t) \mid 1 \leq i < j \leq t\}.$$

In (Marinari et al. 1993) it is observed that, using a rewriting procedure, a set

of generators of syzygies of G can be computed also from all the identities of the following type:

$$x_\alpha b(x_\beta m) - x_\beta b(x_\alpha m) = -\sum_{T \notin LT(G)} c_T x_\beta T + \sum_{T \notin LT(G)} d_T x_\alpha T, \tag{1}$$

where m is an independent term, x_α is different from x_β and there are two polynomials f_i and f_j such that $x_\alpha x_\beta m = T_{i,j}$, x_α divides $LT(f_i)$ and x_β divides $LT(f_j)$. From the proof of Lemma 13.1 of Marinari et al. (1993) it follows that $x_\alpha m$ and $x_\beta m$ are dependent terms. As in (Cioffi 1996, 1999), we will employ the observation of (Marinari et al. 1993) to design a minimalization procedure having always polynomial cost and applicable to each homogeneous polynomial ideal.

Since G is a Gröbner basis, it is not surprising that, in this context, we prove statements using a rewriting procedure. Let

$$\mathcal{V}_d \;=\; \{x_0, \ldots, x_r\} \cdot \bar{B}_{d-1} = \{x_i f_l \mid f_l \in \bar{B}_{d-1}, i = 0, \ldots, r\} = \{p_1^*, \ldots, p_q^*\}.$$

Recall that, if f_l belongs to $\bar{B}_{d-1}$, then $LT(f_l)$ has an independent predecessor.

LEMMA 3.1: *Let f be a homogeneous polynomial of degree d. If any term appearing in f has an independent predecessor, then f can be described as a K-linear combination of polynomials in $G_d \cup \mathcal{V}_d$.*

Proof: By the hypothesis, $LT(f)$ has an independent predecessor, i.e. there exist a variable x_k and an independent term H such that $deg(H) = d - 1$ and $Hx_k = LT(f)$. If $LT(f)$ does not have a dependent predecessor, there exists a polynomial f_i of G_d such that $LT(f) = LT(f_i)$. Then we consider $F_1 = f - f_i$ in the place of f. If $LT(f)$ has a dependent predecessor, there is a variable x_i, $i \neq k$, such that $(H/x_i)x_k$ is dependent. Notice that, since H is independent, H/x_i must be independent too. Thus, there is a polynomial p of $\bar{B}_{d-1}$ such that $LT(p) = (H/x_i)x_k$ and there is a polynomial p^* of $\mathcal{V}_d$ with $LT(p^*) = Hx_k = LT(f)$. Hence we can consider the polynomial $F_1 = f - p^*$ in the place of f. If $F_1 = 0$, the proof ends. Otherwise, by construction F_1 satisfies the hypothesis and $LT(F_1)$ is lower than $LT(f)$. Since $<$ is a term order, after a finite number of steps we must obtain the null polynomial so that f must be equal to a linear combination of polynomials of $G_d \cup \mathcal{V}_d$. □

PROPOSITION 3.1: *Let $f_{x_\alpha x_\beta m} = x_\beta b(x_\alpha m) - x_\alpha b(x_\beta m)$, where m is an independent term. Then:*

$$f_{x_\alpha x_\beta m} = \sum_{f \in G_d} c_f f + \sum_{i=1}^{q} d_i p_i^*. \tag{2}$$

Proof: The thesis follows by Lemma 3.1 since by formula (2) the polynomial $f_{x_\alpha x_\beta m}$ satisfies the hypothesis of such lemma. □

We point out that the cited observation of Marinari et al. (1993) about syzygies means that a polynomial f_i of G depends if and only if, its dependence is shown by a syzygy computed from an identity of type (2) by a rewriting procedure. Therefore, we can state the following result.

PROPOSITION 3.2: *A polynomial f_i of degree d of the reduced Gröbner basis G depends on $G-\{f_i\}$ if, and only if, it depends linearly on polynomials of the set $(G_d-\{f_i\})\cup\mathcal{V}_d$.*

Proof: The "if" implication is obvious. Notice that the polynomials $x_\beta b(x_\alpha m)$ and $x_\alpha b(x_\beta m)$ of formula (2) belong to $\mathcal{V}_d$, where m is an independent term of degree $d-2$. By definition of syzygy, f_i depends on $G-\{f_i\}$ if, and only if, the i-th component of a syzygy computed from an identity of type (2) is constant. From proposition 3.1 it follows that this dependence is equivalent to the existence of an identity of type (2) with a nonzero coefficient c_{f_i}, since formula (2) has been obtained by a rewriting procedure. □

This result allows us to formulate the following procedure to minimalize G. If α is the minimum degree of the polynomials of G, it is easy to observe that the polynomials of G_α are already independent. Hence, for each degree d such that $\alpha+1\leq d$ and $|G_d|\neq 0$, we consider a matrix M_d whose rows correspond to the terms of degree d which are multiples of independent terms of degree $d-1$, i.e. to the leading terms of the polynomials of $\bar{B}_d$.

The first columns of M_d are computed in the following way. For each polynomial p of $\mathcal{V}_d=\{x_0,\ldots,x_r\}\cdot\bar{B}_{d-1}$, we distinguish two cases: if the polynomial p has a leading term with an independent predecessor, then we put the column of its coefficients in M_d; otherwise, if we have already considered a polynomial p' of $\mathcal{V}_d$ such that $LT(p')=LT(p)$ does not have an independent predecessor, we put the column of coefficients of $p-p'$ in M_d. The last columns of M_d are the vectors of coefficients of the polynomials of G_d.

To single out minimal generators of I in G it is now enough to compute the row-echelon form of M_d. In fact, by proposition 3.2 the polynomials of G_d which are minimal generators correspond to columns of M_d that contain a pivot of the row-echelon form of M_d. Moreover, we can compute how a dependent polynomial f of G_d depends on $(G_d-\{f\})\cup\mathcal{V}_d$.

Remark: It is evident that in this context we need to compute the polynomials of $\bar{B}_{d-1}-G_{d-1}$ and, hence, the normal form of their leading terms. In general there is much interest in the computation of normal forms and there are several methods, the convenience of which can depend on the context (see, for example, Mourrain (1999)). If the Gröbner basis has been constructed by the Buchberger and Möller algorithm, then also the normal forms can be computed by this algorithm. Otherwise, by the scheme of Faugère et al. (1993), in (Marinari et al. 1993) it is already suggested a recursive procedure that is good for our algorithm and that is based on the following formula: if $T=x_iH$, where H is a dependent term and $NF(H)=\sum_j c_j x_j T_j$ $(c_j\in K)$, then $NF(T)=\sum_j c_j NF(x_jT_j)$.

Let M be a graded submodule of the graded S-module S^t. For M it is possible to give the definitions of graded term order and of Gröbner bases analogous to the corresponding definitions for the ideal I. So we can define also the corresponding sets $\bar{B}_d^M$ and $\mathcal{V}_d^M$ for M, and it is easy to formulate Lemma 3.1 and Proposition 3.1 for M too. As a consequence, the following statement is obvious, making possible the application of the algorithm to any module of the type of M.

PROPOSITION 3.3: *Let $G^M = \{f_1 = (f_{1,1}, \ldots, f_{1,t}), \ldots, f_h = (f_{h,1}, \ldots, f_{h,t})\}$ be a reduced Gröbner basis of M. Then an element f_i of degree d depends on $G^M - \{f_i\}$ if and only if it depends linearly on the elements of the set $(G_d^M - \{f_i\}) \cup \mathcal{V}_d^M$.*

4. An improvement

The improvement we propose concerns the matrix M_d, the construction of which has been described in section 3. We realized that it is sufficient to consider a submatrix $\bar{M}_d$ of M_d determined only by the columns corresponding to the polynomials obtained by $\mathcal{V}_d$ and only by the rows that correspond to all the dependent terms of degree d that have an independent term as predecessor. Indeed, with the notation already fixed, the following statement holds.

PROPOSITION 4.1: *Let R be the matrix obtained by a Gauss reduction of the transpose of $\bar{M}_d$. Let G'_d the subset of G_d consisting of the polynomials of G_d the leading terms of which correspond to columns of R not containing a pivot. Then $\cup_d G'_d$ is a set of minimal generators of the ideal I.*

Proof: First of all, we observe that a pivot of the matrix M_d can occur only in correspondence of a dependent term because by construction it is the leading term of a polynomial of I. Hence, the pivots of M_d are in the same position as the pivots of $\bar{M}_d$, and as a consequence the rank of M_d is equal to the rank of $\bar{M}_d$. This shows why we can avoid considering the independent terms.

Then, we note that since the rows of the transpose of $\bar{M}_d$ correspond to the polynomials and the columns correspond to the terms, a Gauss reduction of this matrix cannot change the position of the terms with respect to the columns. By Proposition 3.2 it follows that in order to have only minimal generators we can remove from G_d the polynomials corresponding to the rows of the transpose of $\bar{M}_d$ that vanish by the Gauss reduction. It is evident that the rows corresponding to the polynomials of G'_d cannot vanish and, hence, that such polynomials are minimal generators. To see that the polynomials of $G_d - G'_d$ are dependent, it is sufficient to apply a rewriting procedure such as in the proof of Lemma 3.1. □

Applying the above result we save space memory and time for several reasons: (a) considering the transpose of $\bar{M}_d$ allows one to save memory by storing only the rows containing a pivot; (b) the number of columns (resp. of rows) of $\bar{M}_d$ (resp. the transpose of $\bar{M}_d$) is lower than the number of columns of M_d; (c) to decide if the polynomials of G_d are or are not minimal generators we need not reducing them with respect to the transpose of $\bar{M}_d$.

5. The algorithm and some tests

On the basis of the results of sections 3 and 4, we can now outline the fundamental steps of an algorithm for minimalizing reduced Gröbner bases of homogeneous polynomial ideals and evaluate its computational cost.

Algorithm `MinBase`

Input: the reduced Gröbner basis $G = \{f_1, \ldots, f_t\}$ (w.r.t a graded term order) of a homogeneous polynomial ideal I of $S = K[x_0, \ldots, x_r]$.
Output: a minimal set of generators of the ideal I.

begin
$a := min\{deg(f_i) \mid f_i \in G\}$, $b := max\{deg(f_i) \mid f_i \in G\}$
$d := a$
repeat

$d := d + 1$

Computation and sorting of the terms of degree d that have at least an indepedent predecessor

$H_{S/I}(d) := |\{T \in \mathcal{T}_d \mid T \notin LT(G)_d\}|$

for $T \in LT(G)_d$ with at least an independent predecessor **do**

computation of $NF(T)$ (by the scheme of Faugère et al. (1993) cited in the remark of section 3)

endfor

Computation of the transpose of the matrix $\bar{M}_d$ as described in section 4

Gauss reduction of the transpose of $\bar{M}_d$ and identification of the dependent polynomials of G_d by Proposition 4.1

until $d = b$
end

Let $h = max\{H_{S/I}(d) \mid a \leq d \leq b\}$. Note that every polynomial $b(T)$ of degree $d-1$ has at most $H_{S/I}(d-1)+1$ non null coefficients because the independent terms of degree $d-1$ are $H_{S/I}(d-1)$. Hence, since the polynomials of type $b(T)$ of degree $d-1$ are at most $(r+1) \cdot H_{S/I}(d-2)$, then the computational cost of the normal form $NF(T)$ of T is of order $(r+1) \cdot h^2$. Since the terms T of which we have to compute the normal form are at most $(b-a) \cdot (r+1) \cdot h$, then the computation of the normal forms is of order $(b-a)r^2h^3$.

Although the transpose of $\bar{M}_d$ has at most $(r+1)h$ rows and columns, its construction and Gauss reduction are equivalent to a reduction of a matrix with at most $(r+1)^2 H_{S/I}(d-2)$ rows (note that $(r+1)^2 H_{S/I}(d-2)$ is an upper bound for the number of polynomials of $\mathcal{V}_d$). Hence the computational cost of the construction and of the reduction of the transpose of $\bar{M}_d$ is of order $O(h^3 r^4)$. Thus the described method of minimalization is an $O((b-a)h^3r^4)$ algorithm.

Note that the sorting of the terms needs no more than $h^2(r+1)^2$ comparisons at each degree.

The above algorithm has been implemented in a program called `minbase` in C++ compiled using g++ and NTL 5.2 (Shoup 2001) for reduced Gröbner bases w.r.t. the graded reverse lexicographic (deg-rev-lex) term order, because this order is the best suited for minimalization (Bayer Stillman 1987). Our program is available at `http://cds.unina.it/~cioffifr` and, for particular applications, in a software called `Points` (Orecchia et al. 2001). We have tested its performance on randomly generated rational curves of degrees $d = 40, 50$ in $\mathbb{P}^r$ with $r = 4, 6, 8, 10, 12, 14, 16, 18, 20$.

In the following table we report the results of our tests: r is the dimension of the projective space in which the curve is embedded; d is the degree of the curve; `time` is the timing of the performance of our program in seconds; `size` is the memory space used during the computation in kilobytes. The inputs for these tests are available at `http://cds.unina.it/~cioffifr`. All the computations are performed on $K = \mathbb{Z}_p$ with $p = 31991$, on an Intel Pentium IV 1.6 GHz with 512 MB RAM +240 MB swap, running Linux (kernel 2.4.3).

smooth rational curves

r	d	time	size	d	time	size
4	40	0.19	800	50	0.40	940
6		0.32	820		0.49	996
8		0.53	824		0.52	976
10		0.37	1016		0.86	1100
12		0.55	924		0.80	1384
14		1.63	1104		1.23	1356
16		1.54	1216		4.20	1564
18		1.51	1480		3.94	1924
20		1.55	1316		3.77	2072

Notes

This paper is dedicated to Bruno Buchberger in occasion of his sixtieth birthday.

References

Abbott, J., Bigatti, A., Kreuzer, M. Robbiano, L. (2000), 'Computing ideals of points', *J. Symbolic Computation* **30**(4), 351–356.

Abbott, J., Kreuzer, M. Robbiano, L. (2001), Computing zero-dimensional schemes. Preprint available at http://cocoa.dima.unige.it/research/publications.html.

Albano, G., Cioffi, F., Orecchia, F. Ramella, I. (2000), 'Minimally generating

ideals of rational parametric curves in polynomial time', *J. Symbolic Computation* **30**(2), 137–149.

Bayer, D. Stillman, M. (1987), 'A criterion for detecting m-regularity', *Invent. Math.* **87**, 1–11.

Berry, T. G. (1997), 'Parameterization of algebraic space curves', *J. Pure Appl. Algebra* **117/118**, 81–95. Algorithms for algebra (Eindhoven, 1996).

Berry, T. G. (2000), 'Groebner bases of the ideal of a space curve', *J. Pure Appl. Algebra* **148**(1), 17–27.

Buchberger, B. (1998), Introduction to Gröbner Bases, *in* 'Gröbner Bases and Applications', Vol. 251 of *London Mathematical Society, LNS*, Cambridge University Press, pp. 3–31.

Caboara, M., Kreuzer, M. Robbiano, L. (2002), Minimal Sets of Critical Pairs. Available at http://cocoa.dima.unige.it/research/publications.html.

Chiantini, L., Cioffi, F. Orecchia, F. (2001), Computing minimal generators of ideals of elliptic curves, *in* 'Applications of algebraic geometry to coding theory, physics and computation (Eilat, 2001)', Kluwer Acad. Publ., Dordrecht, pp. 23–35.

Cioffi, F. (1996), Calcolo di generatori di ideali di punti e curve algebriche, PhD thesis, Università di Napoli "Federico II". Preprint n. 23, Dip. di Matematica e Applicazioni "R. Caccioppoli".

Cioffi, F. (1999), 'Minimally generating ideals of points in polynomial time using linear algebra', *Ricerche di Matematica* **XLVIII**(1), 55–63.

Cioffi, F. Orecchia, F. (2001), Computation of minimal generators of ideals of fat points, *in* 'ISSAC 2001', ACM (Association for Computing Machinary), pp. 72–76.

Faugère, J. C., Gianni, P., Lazard, D. Mora, T. (1993), 'Efficient computation of zero-dimensional Gröbner bases by change of ordering', *J. Symb. Comp.* **16**(4), 329–344.

Kreuzer, M. Robbiano, L. (2000), *Computational Commutative Algebra 1*, Springer.

Macaulay, F. S. (1927), 'Some properties of enumeration in the theory of modular systems', *Proc. London Math. Soc.* **26**(2).

Marinari, M. G., Moeller, H. M. Mora, T. (1993), 'Gröbner bases of ideals defined by functionals with an application to ideals of projective points', *AAECC* **4**, 103–145.

Mourrain, B. (1999), A new criterion for normal form algorithms, *in* 'Applied algebra, algebraic algorithms and error-correcting codes', Vol. 1719 of *LNCS*, Springer, pp. 430–443.

Orecchia, F. (2001*a*), 'The ideal generation conjecture for s general rational curves in $\mathbb{P}^r$', *Journal of Pure and Appl. Algebra* **155**(1), 77–89.

Orecchia, F. (2001*b*), 'Implicitization of a general union of parametric varieties', *J. Symbolic Computation* **31**(3), 343–356.

Orecchia, F., Cioffi, F. Ramella, I. (2001), *Points (software for computations on points)*, Available for linux platform at `http://cds.unina.it/~orecchia/gruppo/EPoints.html`.

Ramella, I. (1990), Algoritmi di Computer Algebra relativi agli ideali di punti dello spazio proiettivo, PhD thesis, Università di Napoli "Federico II". Preprint n. 30, Dip. di Mat. e Applic. "R. Caccioppoli", 1990.

Shoup, V. (2001), *NTL: a Library for doing Number Theory*, Open source software distributed under the GNU General Public License and available at `http://www.shoup.net/ntl`.

Naive Axiomatic "Mengenlehre" NAM for Experiments
(dedicated to Bruno Buchberger for his 60-th Birthday)

Werner DePauli-Schimanovich[1]

(0) Abstract[2]

Mathematicians still use Naive Set Theory when generating sets without danger of producing any contradiction. Therefore their working method can be considered as a consistent inference system with an experience of over 100 years. My conjecture is that this method works well because mathematicians use only those predicates to form sets, which yield decidable and consistent predicate extensions. And for every open formula they use in the process of constructing of a certain (special) set, we can always find an "almost-closed" formula (i.e. one with only the free variable "x") which yields the same certain (special) set as predicate extension as constructed in the process before. Therefore the use of predicates with free parameters in the Comprehension Scheme does not cause any difficulties.

KEYWORDS: naive set theory, Quine, new foundation, NF, universal sets, comprehension schema, predicate extension, philosophy of set theory, Zermelo, Fraenkel, ZF, complement.

(1) Introduction

When David Hilbert delivered his famous talk at the 1928 International Congress of Mathematicians in Bologna, he claimed: "Cantor has created a Paradise for us from which nobody can expel us again!" This paradise, the Naive "Mengenlehre", was based mainly on a single principle, which was taken as a dogma: "Sets are Predicate-Extensions." Or in other words: "For every arbitrary predicate, its extension (that is, all objects with this property) forms a set." After formalization this principle is called the general Comprehension Scheme (CoS) := forall wff A: forall y: [y in {x: A(x)} <==> A(y)].
wff is the abbreviation for well-formed formula.

In 1903, Bertrand Russell discovered the paradox[3] named after him, to wit, that the predicate "x non-in x" is a substitution-instance for A which falsifies the comprehension scheme. He wrote a letter to Gottlob Frege and told him his discovery. Frege had just finished his book "Grundlagen der Arithmetik" in which he wanted to put arithmetic and "Mengenlehre" onto a safe and sound fundation. He was forced to add an appendix: "Nothing worse can happen to a scientific writer than when his fundation breaks away after finishing his house."

Later other inconsistencies were found: circa 1908 Burali-Forti's paradox of the "set of all ordinals", circa 1909 Mirimanoff's paradox of the "set of all sets not containing an infinite descending element sequence", etc. In all these cases one single formula A was enough to falsify (CoS). Therefore set theoreticians posed themselves the question whether one should not restrict set formation to the "consistent" predicate-extensions (with free parameters); i.e. only accept those (non-pathological) A which, when substituted in (CoS), do not produce a contradiction. But soon it turned out that neither the complement "ko(m)"[4] of an arbitrary set "m" nor the Anti-Russell "x in x" allows one to deduce a contradiction if they are substituted into (CoS); but assuming both together simultaneously does so.

---

[1] Institute for Statistics and Informatics, University Street 9, A-1010 Wien / Austria. Phone: +43.1-4277-386.12, Fax: +43.1-4277-9386, http://www.univie.ac.at/bvi/europolis, Werner.Schimanovich@univie.ac.at

[2] I want to thank Martin Goldstern, Randall Holmes, Matthias Baaz and Thomas Forster for their help and support by writing this paper.

[3] In a forthcoming paper I analyze the concept "paradox" in which I show that it can be split up into the 3 different notions "pathological", "antinomical" and "abnormal", each of which can be characterized formally.

[4] In the following we will always use K as abbreviation for complement [as the German word "Komplement"], because the letter C is already used for Comprehension, Church, Cardinal, Conditional, Cofinal, etc.

Therefore experts in set theory asked themselves whether the closed consistent predicate-extensions in (CoS) would yield a contradiction-free axiomatization of Naive "Mengenlehre". But this restriction was not enough. Because if you take some arbitrary undecideable formula[5] UD together with the implication of the Russell-condition "x non-in x", then its negation non-UD together with the implication of Russell is also undecideable. Each of these 2 implications substituted for A in (CoS) gives a consistent predicate-extension, but either UD or non-UD must be valid and this therefore produces the Russell-class as a set; a contradiction once again.

Now the ultimate question arises: do the "closed decidable consistent predicate extensions", in short CDC-PE (together only with the common axiom of extensionality (EE) and no other specific axioms) produce a contradiction-free collection of formulas?[6] I conjecture: "yes!", and until now nobody could show the contrary. But I was informed by Randell Holmes that it would be better to demand hereditary-consistent[7] formulas A instead of decideable ones. If this system of set theory is consistent, it is probably some version of positive set theory. This will give rise to a series of interesting investigations. Therefore we redefine "consistent" as "hereditary-consistent" what means that decidability as condition can possibly be dropped if it turns out that hereditary-consistent is already enough.[8]

(2) Separation Principle and Limitation of Size

At the time when this ultimate question arose (whether all CDC-PE's together establish a consistent domain of formulas), its investigation was already too late, because the fact that such predicate extensions have to be closed has been denied. Nobody wanted parameter-free set-existence schemata. Georg Cantor had discovered in 1899 the antinomy named after him, i.e. that on the one hand the universal set "us" is of smaller power then its power-set "p(us)": [Card(us) < Card(p(us))], but on the other hand the power-set would have to be extensionally equal to the universal set: [us = p(us)]. This is a classical contradiction, because there exists of course a function from "us" onto "us" (i.e. the identity-function).

This antinomy follows directly from the theorem of Cantor, which says that every set is of smaller power than its power-set. But the theorem of Cantor can be applied to the universal set only because the „Aussonderungs-Prinzip" (= principle of separation) for Cantor was something like an "allgemeingültiges" (generally valid) axiom for arbitrary sets! Therefore it would not have made any sense to consider the closed decidable consistent predicate extensions, because the set theoreticians needed parameters like those used in the separation-principle.

Today the most common separation principle is the axiom-scheme (Sep):
forall wff A: forall x: Set({y: y in x & A(y)}). Or in another formulation:
exist y: forall x: [y = (x intersection A)], where A is the class {|z : A (z)|} of all objects z with A.[9]
If we want to accept the principle of separation as a generally valid axiom for arbitrary sets (which can occur as free parameters), then the „Limitation of Size" ideology follows directly. Therefore Hausdorff and likewise Fraenkel wrote in their fundamental textbooks (in the 1920s): "The antinomies arise through the construction of sets which are extensionally equal with the universe!" Based on this ideology is the axiomatic system of set theory (today most well-known), Zermelo-Fraenkel (= ZF), and adding the axiom of choice (AC), the system ZFC.

---

[5] A closed wff A shall be decidable in respect to FOL (with the only predicate "in") and (EE). If A is almost-closed (with only free x) we call it decidable if all its substitution instances with closed terms are decidable. Any PE shall be called decidable if its substituted A is it.

[6] This collection needs not to be recursively enumerable and therefore no (classical) axiom-system.

[7] Hereditary-Consistent (= HC) means: Restrict comprehension to the formulas with the property that all subformulas (suitably defined) are consistent (i.e. non-pathological). This yield the CHC-PE if we drop D.

[8] Concerning set-existence the CDC-PE's are a "maximal consistent set of formulas" (in a certain sense), and the corresponding sets form a "maximal model".

[9] In class theory (Sep) is the subset axiom: Every subclass of a set is again a set: forall X: [X =< y ==> Set (X)].

Today this ”Limitation of Size“ ideology cannot be kept any longer. In 1937 Willard Van Orman Quine published his system ”New Foundation“ (= NF)[10], which he developed from Russell's theory of types. This system contains a universal set and must thus be inconsistent in virtue of Fraenkel. But in 1969 Ronald Björn Jensen showed the relative consistency of NF with ZFU (= ZF with Urelemente). Since the days of Zermelo-Fraenkel a lot of set theoretical systems with a universal set have been developed in the meantime. A beautiful collection of these systems can be found in the book of Thomas Forster.

(3) Small Sets, Complementation, and ZF

In my dissertation, “Extension der Mengenlehre“ 1971, and its popular-scientific edition “Der Mengenbildungs-Prozess“ (1971 too, in the journal for literature „Manuskripte ’33“) I had raised the question as to a consistent axiomatization of the Naive Mengenlehre. At that time I could present only partial results, e.g. the answer to the question ”What is the slightest modification of ZF needed to add a complement-axiom?“ My answer was: the restriction of the axiom of separation to “small” sets. Of course, the axiom of replacement -- which says, formulated in the language of class theory, that “Every image of a set is again a set“ -- has to be restricted too. (In fact it's enough to restrict replacement, because separation is a logical consequence of it.)

The vague notion of a “small” set can be made precise in several ways:
(1) Small shall be represented by well-foundedness, or hereditary foundedness,
(2) The Cantorian sets (for which the theorem of Cantor is valid) are the small ones,
(3) “Slim” or “Komplements-schmächtiger“, i.e. the small sets are those with smaller power than their complement.
In the simple language of class theory, the restricted axiom of separation says: „The subclass of every small set is again a set.“ The analogue is valid for the images. No mathematician would ever try to construct a subset or image of the universal set, or of some other set “extensionally equal to the universe“. In any case, the mathematicians always do the right thing!

If we restrict the axiom of replacement to small sets (and from this follows already the restricted axiom of separation), then this is (at least for me) the slightest modification of ZF which allows us to add the existence of a general complement. In 1971 at the International Logic-Congress in Bukarest I gave a talk on this subject and explained it to Dana Scott (and after that I discussed it with him in Vienna). At that time I even hoped to obtain a proof of the relative consistency of my system with ZF. But Martin Goldstern convinced me 5 years ago that this proof is false. (And good-minded as I am, I did believe him.) He published another proof for a stronger weakened ZF on his homepage.

After all which has been said until now, one fact should have become evident: If there exists a universal set or the complement in a set theoretical system, then the axioms of replacement and separation have to be restricted to small sets. Cantor wanted the general separation, but that guided us directly into the dead-end street of the Limitation of Size Ideology and therefore to ZF. Despite the fact that ZF is the favorite system of logicians, nevertheless, it is highly unnatural. No mathematician constructs his sets bottom-up by iterated application of the ZF-axioms!

But this was not the reason why ZF was created. Concerning Hilbert's Program the scientific community wanted to show the consistency of set theory. But since Kurt Gödel we know that this is impossible, because (for a sufficiently rich system) one cannot show its consistency by means of the system itself. Therefore ZF has lost its original justification, and it is today in common use mainly because it has been spread all over the world and it is somehow easy to handle (compared e.g. with NF). Mathematicians (especially Bourbaki)

[10] A good introduction to NF the reader can find in Holmes [1998].

have in any case always ignored ZF, and they are still working with the Naive “Mengenlehre”, which they use contentedly without generating any contradictions.

(4) How do Mathematicians Generate Sets?

Taking the logical point of view which I have described just above (i.e. if somebody accepts the restriction of replacement and separation), naturally the question arises how the working method of the mathematicians can be justified philosophically? (ZF certainly does not do the job!) Therefore I shall investigate here the methods mathematicians actually use when they are generating sets.

Working mathematicians have a very strong intuition. And this intuition gives them the insight that a formula A does not produce a contradiction in the comprehension scheme (CoS) even when it is very complicated. They also use only hereditary-consistent formulas and never undecidable ones. (Since these formulas are usualy very large [like the Gödel-formula in Gödel’s proof or the theorem of Paris-Harrington] mathematicians never use them in practise). Of course, mathematicians often use open decidable and hereditary-consistent predicate extensions with free parameters, e.g. for constructing the union or intersection of finitely many sets, or n-tupels. But this problem can be lifted once we agree that the CDC-PE’s (together with extensionality axiom (EE), that two sets containing the same elements are equal) can be considered to be known as contradiction-free by experience. Then the use of such open extensions can be shown to be correct by meta-theorems stating that in such cases, a corresponding CDC-PE can also be constructed which forms the same set.[11]

Mathematicians only make relative complements (e.g., in a sigma algebra). Therefore they will not feel disturbed if it were allowed to construct absolute complements of small sets, too. This can already be deduced from the CDC-PE’s yet. Also the axiom of choice is applied by the mathematicians only to small sets. But a restricted (to small sets) axiom of choice (SmallAC) is also compatible with Quine’s NF, and this gives rise to (and hope for) the assumption that it will also be consistent together with the CDC-PE plus (EE).

Therefore if mathematicians take a set-constituting property A and substitute it into the set operator, it always produces a set, and no contradiction. Because mathematicians only use decidable and hereditary-consistent formulas A (which make the PE’s also as a collection consistent), no contradiction in practise arises. If they use formulas A, with only x as free variable, substituting them into the set operator, it makes our philosophical justification easier. But if they use also formulas with free parameters in turn, it does not create difficulties as long as they use only small sets (where the pathological ones are excluded automatically). This can be considered as a 100-year-long experience of consistency of naïve set theory in practice! [12](That is several years more than ZF or NF exists.)

(5) The Systems NAM* and NACT*

The collection of all closed decideable-consistent (or only hereditary-consistent) predicate extensions (= CDC-PE or CHC-PE) together with (EE) form a new system of set theory. But since this collection of formulas is not recursively enumerable, it is not a axiom system in the classical sense. Anyway, the property of a formula A to establish a CDC-PE is easy to grasp, and therefore it does not matter whether the set of all CDC-PE’s is not recursively enumerable, because it is enough to consider some large recursively enumerable subset of it. Let us therefore call this system of all CDC-PE’s plus (EE) an axiom-system in the wider sense; we want to name it NAM* (= Naïve Axiomatic Mengenlehre Star), or if we

[11] A great help in this case would be if we add the term-consistency rule: If for every closed term t of a wff A the statement A(t) holds, then “forall x: A(x)”.

[12] There must exist an adequate formalization of the contentually working method of the mathematicians using naïve set theory. The question is only: Who will find it first? Of course: implicitely I conjecture in this paper: The CDC-PE’s (or the CHC-PE’s) are this adequate formalization of mathematical practise.

formalize it in the more simple language of class theory, NACT* (= Naive Axiomatic Class Theory Star, or: the naked star).

NAM* works in the following way: First we want to find the wffs A with only one free variable x which itself (and all its subsets) do not produce a contradiction substituted into (CoS). Let us therefore consider a formula generator FG which enumerates all wffs Aj in a natural way. For all Aj call their substitution instances (CoS-Aj). A decision algorithm DA shall find out if (CoS-Aj) together with (EE) produces a contradiction only by use of the rules of pure logic (with the only predicate "is an element of") and without any special axioms of set theory at all! If this happens we want to call such a predicate Aj "pathological".[13] Therefore the hereditary-non-pathological Aj's and [if decidable too for short:] logical Aj's are exactly the CDC-PE's. NAM* is therefore (LogoCoS), i.e. (CoS) restricted to the (hereditary-)non-pathological (and if necessary decidable) Aj's, plus (EE).
As a formula: NAM* := (LogoCos) & (EE).

For our purpose (as already mentioned) it does not matter that such a decision algorithm DA does not exist for all (hereditary-)non-pathological predicates. It's enough to find one for a sufficiently large class of non-Patho, which should signify the (second order class of all) pathological classes. (The existence of such a DA for all (CoS-Aj), with the intention of producing the whole class Logo [= non-Patho and decidable], is logically equivalent to the decision problem of FOL, shown to be impossible by Alonzo Church 1936.) The analogue statement is valid also for NACT* (= the naked star): classes with non-patho (and of course decidable) wffs (i.e. the CDC-PE's) are sets.

Therefore we can conclude that NAM* (i.e. the most important system instantiating NAM) is a real naïve set theory, very similar to the one used by mathematicians in practice, with a unique logical principle to generate sets (on which it is based), in contrast to other systems with more or less arbitrarily selected axioms. This shows also that Logicism is not dead yet. NAM* is a rectification of mathematical practise, and the systems ZF and NF are (in a certain meta-mathematical sense) included in NAM*. (Of course we have to consider in this case the restricted axioms of separation, replacement and choice, which have to be shown as meta-theorems, and consider only the decidable statements.)

Set theory was already nearly dead, after all the ZF people had done with it. Dana Scott said on the occasion of receiving the Bolzano award in Prague: "Bernhard Bolzano would have been very disappointed if he had known what excesses set theory of today is producing!" Now, after creating NAM* and NACT*, it will become interesting again. Because Naïve Axiomatic "Mengenlehre" in general (which we want to call short hand NAM in the following) delivers in addition an instrument to experiment with set theory. This way we can find out what mathematicians consider to be normal sets. This is a philosophical program like the one in the '30s of the past century to formalize the intuitive notion of computability. NAM* is only the most significant system found by this experimentation. In the following sections we want to show how experimentation with NAM works.

(6) The System NAM of Systems NAMix

The main goal of NAM is to find a more or less adequately explicit criterion that precisely formalizes the intuitive notion of a "normal set". NAM is mainly a construction procedure for building several formal systems NAMix, each of which can turn out to be an adequate codification of the contentual naive set theory. ("i" is a natural number which enumerates the used normal condition, and "x" is a letter which points to the variants of the used axioms.) Parallel to NAM, the Naïve Axiomatic Class Theory NACT is constructed as a system of systems too.[14]

[13] Of course this DA proves also if non-(CoS-Aj) produces a contradiction, and therefore if Aj [or better its corresponding PE] is satisfyable or not.
[14] In another forthcoming paper, the most important systems of NACT are explained.

In NAM any arbitrary formula can be used as a starting point for the investigation of the "normality" of sets. Once a predicate has been found that is equivalent to the predicate atom "Normal" (and which does not falsify NAM of course) then the concept of a normal set is characterized correctly and uniquely.[15] The tack taken here is similar to attemps of the '30s of the past century to find an adequate explication (or surrogate) for the concept of computability. In NAM one can experiment and study the results and effects of different normality conditions until we have found the most appropriate one. With such a condition mathematicians would finally have a decision criterion for generating sets of the naïve set theory.

In the following we want to suggest a series of criteria as conditions for the characterization of normal sets. If we consider also combinations of them, there will be about 100 criteria. One of them will fit almost certainly. Then mathematicians and logicians can prove formally that a given set is normal. Following this line naïve set theory turns out to be completely free of antinomies or paradoxes once we interpret the comprehension schema correctly. Only the normal sets are predicate extensions! The abnormal ones are e.g. identical to the universal set, or to the empty set, or to some external constant "@", just as we like!

Now let us start with the axiomatic construction of all these systems of NAM:
The only objects in NAM are the sets we want to denote by small Latin letters x, y, z, ... We also use the (usual unary) prime predicate "Normal". The set operator (S-O) produces always only a set (as it is usual in naïve set theory):
{x: A(x)} is a SET !
(And that holds for all A in most systems of NAM.)
But we have to distinguish the set operator from the class operator (C-O):
{|x: A(x)|} is a CLASS !
(Keep in mind that classes are denoted by capital Latin letters X, Y, Z, ... , while the small variables are restricted to sets. But we do not use classes in NAM; and we mentioned this only to avoid confusion! (CoS) for classes cannot produce a contradiction because the pathological predicates A only yield proper-classes and no sets. Therefore we want to call this scheme Church Scheme (CS) for classes, in distinction to (CoS) for sets!

In NAM the general (CoS) is used only in a restricted form. Let "?" be an arbitrary junctor. Then the most important junctive restriction of (CoS) is:
(Ju-CoS) := forall wff A: forall y: [y in {x: A(x)} <==> A(y) "?" Normal] .
"Normal" can also be a structural property of the wff A, e.g.: Normal := CDC-PE (alternatively Logo or hereditary-non-Patho) yields NAM*; Normal := Stratified yields NF; Normal := (A is a ZF-axiom) yields ZF.

(7) The most important Comprehension Scheme (raBaDi-CoS) of NAM

Generally Normal is a unary predicate of FOL with the basic set of (CoS) substituted into its free variable. The most frequent junctor "?" is "or non" added directly after A on the right side of the equivalence. That gives the most important instance of (Ju-CoS):
(raBaDi-CoS) := (y in {x: A(x)} <===> [A(y) or non-Normal({x: A(x)}]).

We want to call this axiom the "rightside-abnormal-Basicset-Disjunction" Comprehension-Schema, shortly: (raBaDi-CoS).[16] With (raBaDi-CoS) the normal sets are predicate extensions as usual. But the abnormal sets inflate (or sometimes even blow up) to the universal set "us := {x: Verum}". They behave somehow like "Urelemente" which are at the

[15] Of course it is also possible to construct a hierarchy of characterizations with increasing strength, excluding cases which are too strongly counter-intuitive.
[16] The author has investigated several restrictions of (CoS) but this one is the best! Of course, one can add Normal also to the left side of the equivalence, or put it as a condition on the entire formula (CoS).

same time sets. (Therefore let us call them set-like urelements.) We have 3 possibilities to handle them:

(A) We identify all pathological sets with the universal set, and the usual axiom of extensionality (EE) works unrestricted as before:
x = y <===> forall z: (z in x <=> z in y) .
But because of substitutivity the universal set has to be abnormal. (But this merely philosophical disadvantage will be considered as a gift by many readers.)
(B) The second possibility would be to restrict the extensionality axiom, similarly to what we do in ZFU (= ZF with Urelemente), i.e. to connect the axiom with the condition "non-Normal(x) or non-Normal(y)". Then one cannot deduce the equality of the Russell set "ru" with the universal set "us", and us can be considered as normal from now on. But this solution complicates the machinery in an unnecessary way.
(C) The third possibility is a restriction of substitutivity. But since this axiom is part of pure logic, this possibility is certainly the least desirable way out (to solve this problem). Therefore we decide to use possibility (A).

(raBaDi-CoS) is so important that we want to investigate it in more detail. "Antinomies arise by construction of sets which are extensionally equal to the universe.", said Fraenkel as we mentioned already. And Jonny von Neumann concluded in the same spirit: "If they are not already equal to the universe, the antinomial sets should be at least of equal power!" (In the new language of class theory they are of course the proper classes and the universal class, respectively.)
(raBaDi-CoS) does more than Jonny wanted. It unifies all abnormal sets with the universal set us := {x: Verum}, what will certainly warm the hearts and enjoy the ZF people. But only for a moment, because (in many systems of NAM) all derivatives of the universal set are normal again, e.g. {x: x =/= us}.

(raBaDi-CoS) is very strong, and together with the axiom of extensionality (EE), a suitable axiom of choice (AC), the set operator (S-O), and some Fundamental- and Eventual-Axioms, it forms a formal system NAM% which is (together with a suitable Normal Condition) similar to naïve set theory and the mathematical praxis (as well as NAM*).

(8) Other Comprehension Schemata of NAM

It is not an accident that (raBaDi-CoS) forms a system of equal strength as NAM%. There exist also other junctive-restricted comprehension schemes of construction series (Ju-CoS) which achieve something similar. For instance, (rinoBaCo-CoS), the "rightside-normal-Basicset-Conjunction" comprehension scheme
y in {x: A(x)} <===> A(y) & Normal({x: A(x)}) .

In the same way as (raBaDi-CoS) inflates the pathological sets to the universal set, (rinoBaCo-CoS) collapses them to the empty set-like urelements. If we once again choose the alternative (A), it turns out that the empty set 0 := {x: Falsum} becomes abnormal, which, to be sure, is unfamiliar but in no way inconsistent. Already the singleton {0} of 0 is again normal (and also everything construed upon it). In order to incorporate the empty set into our consideration of the normal sets, we would have to add into the set operator and the axioms the term "or y = 0" in each case where the condition Normal(y) appears.

(rinoBaCo-CoS) & (EE) & (AC) & (S-O) and the Fundamental- and Eventual-Axioms form again together with a suitable Normal-Condition a similar system as NAM%, which we want to call NAM&. Its logical machinery works exactly parallel to NAM%.
From both mentioned systems NAM% and NAM& we can imagine that it would be enough to select an arbitrary set and let it become abnormal to avoid the antinomies. To arrange this we use this time around an implicative restriction (I-CoS) of (CoS), e.g. the "normal-Basicset-Implication" Comprehension-Scheme (noBI-CoS):
Normal({x: A(x)}) ===> [y in {x: A(x)} <=> A(y)] .

By specification of the antecedence with a suitable Normal-condition, all systems like NAM% and NAM& which are characterized uniquely by a restricted (CoS) like (raBaDi-CoS) and (rinoBaCo-CoS) can be constructed. Let us choose e.g. the Normal Condition "not-Equal-@-Implies-normal-Basicset" (nE@I-noB):
{x: A(x)} =/= @ ===> Normal({x: A(x)}),
where "@" is any arbitrary set (which we want to become abnormal in the antinomic case, e.g.: ru, on, us, 0, etc). (noBI-CoS) again together with the other above mentioned axioms yields also a system, we want to call NAM§. (We can add also the re-implications (CoS-I-noB) and (noB-I-nE@) and study these systems. But this often turns out to be too strong.)

We can also adjoin the "commercial at" "@" as a primitive constant to the language of logic with the effect that then the non-pathological predicate extensions can always form a set and the proper set universe is free from antinomies. Pathological sets are then "aliens" which have nothing to do with set theory any longer.

(9) Basic Axioms of NAM

The Naive Axiomatic Mengenlehre NAM should consist in general of 4 Basic Axioms (BAi), 4 Fundamental Axioms (FAj), some Eventuality Axioms (EAk), and a reasonable Normality Condition (NCl) (selected from the group of normality conditions NC).

The 4 Basic Axioms are:
(BA1) := (S-O). The set operator {x: A(x)} produces a set for each wff A .
(BA2): one of the "normalized" Comprehension Schemata (Ju-CoS) or (I-CoS), e.g.:
(BA2a) := (raBaDi-CoS), or
(BA2b) := (rinoBaCo-CoS), or
(BA2c) := (noBI-CoS), or
(BA2d) := what you like.
In the following we want to concentrate our attention mainly on (raBaDi-CoS).

To the set operator and the scheme we have to add:
(BA3) the Axiom of Extensionality (E), mainly as
(BA3a) in the usual formulation of "Element-Equality" (EE):
forall z: (z in x <=> z in y) ===> x = y,[17]
(BA4) an appropriate formulation of the Axiom of Choice (AC), e.g.:
(BA4a) the Choice-Function Axiom (ACF):
exist f: Function(f) & forall y (=/= empty) Subset x ==> f(y) in y . Or:
(BA4b) the Choice-Set Axiom (ACS):
forall y, z in x: (y =/= empty & y intersection z = empty) ===>
exist u: forall v in x: exist w: u intersection v = {w} . Or:
(BA4c) the Ordinals-Universalset Axiom (On-us), e.g. in the form: On ~ us .
Here "~" is a symbol for equal power, i.e. the existence of a bijection between the 2 sets (or classes). On := {x: On(x)} is the class or set of all ordinal-numbers, and us: = {x: Verum} is the universal set.
(BA4d) some other suitable axiom of choice (e.g. the one restricted to small sets), or maybe none.

(10) Foundamental Axioms of NAM

Furthermore in NAM the following 4 Fundamental Axioms (FAj) should be valid. They express the standard assumptions for normal sets.[18]
(FA1) the Omega Axiom (Om):

[17] Sometimes (as e.g. in the GNAM-systems considered later) we use also (BA3b), i.e. the "Normal-Elements-Equality" (NEE): forall z: [Normal(z) ==> (z in x <=> z in y)] ====> x = y .
[18] The predicate "Normal(x)" cannot become totally "Falsum" because the Fundamental Axioms and some Eventuality Axioms contribute implicitly to the Normal-Atom.

Normal(omega),
(FA2) the Power Axiom (Po):
Normal(x) ===> Normal(p(x)),
where p(x) := {y: y Subset x} is the power-set of x.[19]
Note that not every sub-class Y of x (constructed with the wff A) must also be a subset y of x. E.g.:Ru =< us\{us} & ru = us =/< us\{us} if you use (raBaDi-CoS).
(FA3) the Union Axiom (Un):
Normal(x) & forall y in x: Normal(y) ===> Normal(union(x)),
where union(x) := {y: exist z: y in z & z in x}.
(FA4) shall be a suitable Axiom of Replacement (or Image-set Axiom in the language of class theory) (Im), that the image of a function with a normal domain is also normal. We distinguish different degrees of strength of (Im).[20]

Let us first list the first 3 versions of the image-set axioms:
(FA4alfa) the "Simple" Image-set Axiom (S-Im):
Normal(x) & Functional-Formula(A) & [exist u in x: exist v: A(u, v)] &
y = {z: [exist w: w in x & A(w, z)]} ======> Normal(y).
(FA4beta) the "Normal-or-Null-Elements" Image-set Axiom (NoNE-Im):
Normal(x) & Functional-Formula(A) & [exist u in x: exist v: A(u, v)] &
y = {z: [exist w: w in x & A(w, z)] & [Normal(z) or z = 0]} ======> Normal(y).
(FA4gamma) the "Normal-Elements" Image-set Axiom (NE-Im):
Normal(x) & y = {z: Functional-Formula(F) & [exist u: u in x & F(u, z)] & Normal(z)} =======> Normal(y).

(11) Eventuality Axioms of NAM

After presenting the Basic Axioms (BAi) and the Fundamental Axioms (FAj), we want to formulate now the Eventuality Axioms (EAk)[21]. Some combinations of basic axioms and fundamental axioms (and maybe the used Normal Conditions) it may be clever and wise to add some of these axioms too. E.g.:

(EA1) the "Singleton-Arbitrary-Normal" axiom (SAN):
Normal({@}) , where @ := ru, on, us, 0, etc.
(EA2) the "Equivalent-Formulas-Equal-Sets" axiom (EFES):
[forall x: A(x) <=> B(x)] ===> {y: A(y)} = {z: B(z)} ,
(EA3) the "not-Normal-Equal-Mighty" axiom (nNEM):
non-Normal(x) & non-Normal(y) ===> x ~ y ,
(EA4) the "Elements-Normal-or-Arbitrary" axiom (ENoA):
Normal(x) ====> forall y: [y in x ==> Normal(y) or y = @] .

In addition to these 4 main axioms we can use also:
(EA5) the "Elements-Normal" Eventuality Axiom (EIN-EA) instead of (EA4):
Normal(x) ===> forall y: [y in x => Normal(y)] ,
(EA6) the "Komplement-Normal" Eventuality Axiom (KN-EA), which postulates the normality of the complement of a normal set:
Normal(x) ===> Normal(ko(x)).
(EA7) the "Normal-Set-exclusive-or-Komplement" axiom (NSxorK):
Normal(x) <=/=> Normal(ko(x)).
(EA8) the"Normal-Set-or-Komplement" axiom (NSoK) axiom:
Normal(x) or Normal(ko(x)),

[19] p(x) denotes the power set (of a set or class), while P(X) denotes the power class (of all subsets of a given class), and PC(X) should denote the 2nd order power class of a class. Sometimes it may be more suitable to define p(x) := {y : y Subset x & Normal(y)}, e.g. in the GNAM-systems.
[20] A is here (and also in the following axioms of (FA4)) a Functional-Formula, i.e. a formula with the functional-property (similar to those used in ZF). As abbreviation for this we write F-F(A).
[21] "Eventuality" means case-based in this article.

this axiom plays an important role, but it contradicts (EIN-EA).
(EA9) the "Normal-Set-or-Supplement" (NMoS) axiom is also conceivable:
Normal({x: A(x)}) or Normal({x: Supplement-A(x)}),
where Supplement-A is that formula, where the Epsilon (i.e. the element sign) is replaced everywhere by its negation.
(EA10) Also other similar conditions on wffs A would make sense.

As already mentioned above, the systems of NAM are constructed in general as a combination of 4 groups of axioms: basic, fundamental, eventuality axioms and a normal condition, where we sometimes have to choose the most suitable variant of an axiom. But before we start to consider the first normal conditions (which are in fact only the objects for our experiments to find out the most appropriate formal explication of the notion "Normal") the reader should ask himself: what follows already directly from the axioms of NAM without the normal conditions. We want to call these "conditionless" kernel systems NAM0x, where the "x" symbolizes the construction series (i.e. the variants of the (BA2) axioms). "a" points e.g. for the use of (raBaDi-CoS), "b" for that of (rinoBaCo-CoS), "c" of (noBI-CoS), etc. (NAM0a, b, c are more or less other names for NAM%, &, §.)

(12) The first Normal Conditions and the first Systems of NAM

After this short interruption let us go on with the explanation of the last group: the Normal Conditions (NCl). We want to find the best of these conditions to select. But until we know which condition is the best, we will need some time for experiments. The first two conditions are:

(NC1) the "not-Function-Domain-Komplement" Condition (nFDK-Cond):
non-exist f: (Function(f) & f[x] = ko(x)) ===> Normal(x) .
f[x] is the image of the domain x under f, and ko(x) is the Komplement {y: y non-in x}. By didactical reasons we can add the term "exist g: (Function(g) & g[ko(x)] = x)" in the antecedent, but this term is in fact redundant.

(NC2) the "Smaller-Power-than-Komplement" Condition (SPK-Cond):
card(x) < card(ko(x)) [& x ~/~ ko(x)] ===> Normal(x) ,
where card(x) is the cardinal-number of x, and card(ko(x)) that of the complement. (Again the didactical term in brackets can be omitted.)
(NC1) and (NC2) are essentially only different formulations of one and the same fact.[22]

If we keep the above-mentioned connections in mind, we get first the semi-canonical system NAM1a with (BA2a) := (raBaDi-CoS) plus the other BAi, and (FA4alfa) := (S-Im) plus the other FAj, with condition (NC1). The first 3 EAl are derivable in NAM1a. This system NAM1a will be later the basis for some extensions, e.g. for NAM1aKNoU := NAM1a & (KNoU-EA).

In analogy to NAM1a we get the parallel-semicanonical system NAM1b which is also founded on condition (NC1), but together with (BA2b) := (rinoBaCo-CoS) and (FA4beta) := (NE-Im) and the other axioms (like in NAM1a) plus the 3 eventuality axioms. Like NAM1b we get the quasi-semicanonical System NAM1c with (BA2c) := (noBI-CoS), (FA4beta) := (NE-Im), as well as (NC1), plus the other axioms (inclusive the 3 eventuality axioms). (NAM1c does not imply Cantor's Antinomy.) NAM2a, b, und c, are the same systems as NAM1a, b und c, only based on condition (NC2). NAM2c is a better basis for extensions than NAM2a, yielding NAM2cKN := NAM2c & (KN-EA). Thus, we get (with this correspondence of the

[22] The danger inherent in these conditions can be explained by the following example: Let us define Slim := {x: card(x) < card(ko(x))} and HeredSlim := {x: forall y in x*: card(y) < card(ko(y))}. We want to find out if "Slim in Slim" or "HeredSlim in HeredSlim" ? That would imply an antinomy similar to Cantor's. Fortunately the "Smaller-Power-than-Komplement" Condition (SPK-Cond) and "Hull-Smaller-Mightiness" (HSM-Cond) are consequently wrong for Slim and HeredSlim.

axioms) all together 6 basic systems (in addition to the 3 kernel systems). Note also that NAM2a without foundational axioms, but together with (EA7) := (NSxorK) is equivalent to NACT#. Together with the 4 foundational axioms it is equivalent to NACT#4.

Let us first write down the axioms of the first system NAM1a of the Naive Axiomatic Mengenlehre in a list. NAM1a has 8 axioms:
(1-1) {x: A(x)} is a set,
(1-2a) [y in {x: A(x)} <==> A(y) or non-Normal({x: A(x)})] ,
(1-3) forall z: (z in x <==> z in y) ===> x = y,
(1-4) a suitable axiom of choice, e.g. On := {|x: On(x)|} ~ us,
(1-5) Normal(omega) ,
(1-6) Normal(x) ==> Normal(p(x)) ,
(1-7) Normal(x) & forall y in x: Normal(y) ==> Normal(union(x)),
(1-8) Normal(x) & Functional-Formula(A) & [exist u in x: exist v: A(u, v)] &
y == {z: [exist w: w in x & A(w, z)] & Normal(z)} ====> Normal(y) ,
(1-9) non-exist f: (Function(f) & f[x] = ko(x)) ===> Normal(x) .

Axiom (1-9) is (NC1). The leading "1" points at the 1st NC, because we are enumerating the systems synchronously with the Normal Conditions.
The Eventuality Axioms (1-10) to (1-13) are derivable from NAM1a.
(1-10) Normal({us}) ,
(1-11) [forall x: A(x) <=> B(x)] ===> {y: A(y)} = {z: B(z)} ,
(1-12) non-Normal(x) & non-Normal(y) ===> x ~ y ,
(1-13) Normal(x) ==> forall y: [y in x ==> Normal(y) or y = us] ,
NAM1a produces mainly such normal sets which are similar to the sets of ZFC; but also more than that.

(13) Complements and restricted Image-set Axioms

If we want to add a complement to NAM1a we would put also the "Komplement-Normal-or-Universal" Eventuality Axiom (EA6') := (KNoU-EA) as axiom with the number 14 onto the list:
(1-14) Normal(x) ==> Normal(ko(x)) or ko(x) = us,
Let us call this system NAM1aKNoU, where the pathological sets in the "middle" are abnormal, while small and large sets are normal.[23] But in that case we have to restrict the domain in the Image-set Axioms (1-7alfa, beta, und gamma) to small sets; otherwise it would be possible to deduce the normality of the Russell-set "ru". Therefore (1-8) has to be replaced by stronger restricted axioms (e.g. (FA4delta, eta, fi, psi, chi, jota, or kappa)). Before we can decide which one we want to select let us enumerate the new image-set axioms in a list.

(FA4delta) the "twofold Functional-Formula" Imageset Axiom (2FF-Im):
Normal(x) & y1 ==
== {y: ([Funktional-Formel(F) & Funktional-Formel(G) & (forall x1, y1, u, v: x1 =/= y1 & G(x1, u) & G(y1, v) => u =/= v)] ====> [exist z: z in x & F(z, y) & Normal(y) <==> non-forall w non-in x: exist z in x: G(z, w)] } ========> Normal(y1) .
In fact this imageset axiom is a double formula scheme[24] which expresses the same fact as the following two other formulations:

[23] For NAM1aKN, the same is valid as for ZFK (respectively ZFCK).
[24] For unsuitable F or G, y1 will yield the universal set. (Therefore it would be good to use either (KNoU-EA) or NAM2c as basis.) The right-side expression in the parentheses (in the antecedent of the implication in the set-brackets) is only added for didactical reasons, since everybody knows that G is an injection. In the consequent (i.e. right side of the implication) in the set-brackets, the contra-valent (respectively the equivalence with the negation) is valid! Therefore not both conditions in the 2nd brackets [= square brackets] can simulaneously have the same truth-value. The term on the right side of the equivalence says that G is no surjection of x onto its complement. E.g.: Let x := omega. Also if you can find an appropriate F-F(G) (such that the antecedent becomes true) where G can map omega onto its complement, the 2nd condition of the contra-valent always becomes

(FA4eta) the “not-Function-Domain-Komplement” Image-set Axiom (nFDK-Im):
Normal(x) & y1 ==
= {y: Functional-Formula(F) & (non-exist g: Function(g) & g[x] = ko(x)) & [exist z: z in x & F(z, y)] & Normal(y)} ========> Normal(y1) .
Here g[x] is the image of the domain x of the function g.

(FA4phi) the “Domain-Smaller-Power-as-Komplement” Bildmengen-Axiom (DSPaK-Im):
Normal(x) & y1 ==
== {y: Functional-Formula(F) & [card(x) < card(ko(x))] & [exist z: z in x & F(z, y)] & Normal(y)}
========> Normal(y1) .

The condition slim(x) := [card(x) < card(ko(x))] expresses the fact that x should be “small” in some specific sense. But instead of using this formulation of “small” we can also use another condition, e.g.: Mirimanoff, Founded, Hereditary Founded, or Cantorian, which yields the following axioms. (Note that in NAM it depends of the system chosen whether these conditions are equivalent or not.)

(FA4psi) the “Mirimanoff-Domain” Imageset Axiom (MD-Im) with:
Mirimanoff(x) := non-exist f: [Function(f) & f(0) = x & forall n in omega: f(n+1) in f(n)].
(FA4chi) the “Founded-Domain” Image-set Axiom (FD-Im) with:
Found(x) := exist y in x: x intersection y = empty .
(FA4jota) the “Hereditary-Founded-Domain” Image-set Axiom (HFD-Im) with:
Heri-Found(x) := forall y in e*(x): exist z in y: z intersection y = empty .
(FA4kappa) the “Cantorian-Domain” Image-set Axiom (CD-Im) with:
Cantorian(x) := [card(x) < card(p(x))] .

If we drop the appendix “Normal(y)“ in the curly braces (= set parentheses) of these new 7 axioms, let us call them (FA4d, e, f, g, h, j, und k). This release from “Normal(y)” will allow us to also generate the derivatives of the universal set and similar sets. (In some systems the normality of the elements is redundant because it follows from other facts.) In analogy to NAM1aKN we get further systems like NAM2aKN, etc, if we replace the image-set axiom (1-8) by one of the 7 new (FA4).

To formulate the other NC’s we want to explain what is the meaning of the hull e* of the elementhood relation e := {<x, y>: x in y}). Let us define:
x e* y :<==> exist n (=/=0) in omega: exist f: Funktion(f) & f(0)=x & f(n-1)=y & forall i in n: f(i) in f(i+1) .
e* := {<x, y>: x e* y}, [or the union from n = 0 until omega: e^n] .
It is known that e* can be defined within NBG and ZFC, and -- as one can see -- also within NAM1a, etc. In those systems where that should not be possible, we have to use the class theory CNAM (constructed over NAM) and define the class relations E, E^n and E* in it.

Since with e and E, “e...” and “E...” are also relations, we can fix their ranges as we are accustomed to be “e...(x)” := {y: exist z in x: <z, y> in e...} and “E...(X)“ similarly. In analogy to the above we write for the complements of e^n, E^n, e* and E* the following:
n-e^n, n-E^n, n-e* or n-E*, and the same symbols for the predicates.
e^+ (or epsilon-plus) shall be e* \ id and E^+ shall be E* \ Id.

(14) Further Normal Conditions

With this preliminary work we can proceed with the definitions of the NC’s:
(NC3) is the “Elements-of-Hull-no-Function-from-Domain-to-Komplement” (EHnFDK-Cond):

---

wrong (because that is impossible). Therefore omega is of smaller power than its complement, and therefore the 2nd condition is true: we can produce an image of omega! But if x = ko(omega), then the 2nd condition is true and the 1st one must be wrong: we cannot construct an image of ko(omega).

forall y in e*(x): [non-exist f: Function(f) & f[y] = ko(y)] &
[exist g: Function(g) & g[ko(y)] = y] =====> Normal(x) .
The 2nd term in the brackets in the antecedent is only of didactical nature and can be dropped.
(NC4) the "Hull-Smaller-Mightiness" Condition (HSM-Cond):
forall y in e*(x): card(y) < card(ko(y)) =====> Normal(x) .
This condition is more complicated than (NC2), but it does hinder the further construction of "counter-intuitive" sets.
(NC5) the "no-Bijection-Domain-Komplement" Condition (nBDK-Cond):
non-exist f: (Function(f) & f[x] = ko(x)) or …
... or non-exist g: (Function(g) & g[ko(x)] = x) ====> Normal(x) .
(NC6) the "not-Equal-Mighty-Komplement"[25] Condition (nEMK-Cond):
x ~/~ ko(x) ===> Normal(x),
what means that if x is not equal mighty with its complement then it is normal.[26] Note also that NAM6c without foundamental axioms, but together with (EA8) := (NSoK) is equivalent to NACT+. Together with the 4 foundational axioms it is equivalent to NACT+4.

Just as before we claim now the validity of the last 2 conditions for all elements of the hull of x. Thus we get:
(NC7) the "Elements-of-Hull-not-Bijection-Domain-Komplement" Condition (EHnBDK-Cond):
forall y in e*(x): "the antecedens of (NC5) with (x/y)" ====> Normal(x) .
(NC8) the "Elements-of-Hull-not-Equal-Mighty-Komplement" (EHnEMK-Cond):
forall y in e*(x): y~/~ko(y) ====> Normal(x) .

There exists of course a whole series of further Normal Conditions that generate also more or less interesting systems. E.g.:
(NC9) the "not-Elementship-of-ItSelf" Condition (nEIS-Cond)[27]:
x non-in x ====> Normal(x) .
(NC10) the "not-Element-of-It's-Own-Hull" Condition (nEIOH-Cond):
x non-epsilon-plus x ====> Normal(x),
"x non-in e^+(x)" means: non-exist n =/= 0 in omega: x in x1 in x2 in . . . in xn = x .
(NC11) Mirimanoff-Condition: "not-Infinite-Descending-Element-Sequence" (nIDES-Cond):
non-exist f: [Function(f) & f(0) = x & forall n in omega: f(n+1) in f(n)] ===> Normal(x) .
(NC12) the Foundedness condition (Found-Cond):
exist y in x: x intersection y = empty ====> Normal(x) .

These 4 conditions can again be strengthened by the claim of the validity for all elements of the hull of the former domain:

---

[25] We want to use in the context the notion "Equal-Mighty" instead of "equipollent", because it allows us the easier abbreviation. Furthermore, it represents the new language Euro-English. We need not to speak American English in Europe, and British English is too difficult for us.

[26] Be careful with (NC6) := "not-Equal-Mighty-Komplement" (nEMK-Cond) or (NC8) := "Elements-of-Hull-not-Equal-Mighty-Komplement" (EHnEMK-Cond) ! Because (raBaDi-CoS) and (rinoBaCo-CoS) force "us" or "0" to be abnormal, we should add the appendix "or x = us or x = 0" to "Normal(x)" at the end of (NC5) to (NC8) and call them (NC5') to (NC8'). Or we have to use (noBI-CoS) and "@ := on". And this implies (because of "not-Normal-Equal-Mighty" (nNEM)) the strong axiom of choice (OnAs) := On ~ us .

[27] In some systems (especially those formed with (NC9) etc) e.g. the following is valid: Let ru := {x: x non-in x} be the Russell-set. The conditional Comprehension Scheme (noBI-CoS) together with its re-implication (CoS-I-noB) makes the implication to an equivalence yielding (noBE-CoS). This scheme allows the deduction of the following facts: ru = ru u {ru} = {x: x non-in x v x = ru} . ("u" is here the union, and "v" is the "or".) This is reasonable because the right-hand term of the implication (= the one with the equivalence in (noBI-CoS)) produces a contradiction. This falsifies the antecedent and yields "ru in ru". But at the same time the following is also true: ru = ru \ {ru} = {x: x non-in x & x =/= ru}. You cannot take ru out from itself. The same is valid for the other pathological sets too, like the ordinal-set on, the Mirimanoff-set mi, the Founded-set fu, etc. With ru we have also as many pathological sets as its elements, because we can generate (ru \ {x}) and (ru \ x) for every element x in ru (which are mostly normal sets), and for these new products (ru \ {x}) in (ru \ {x}) and also (ru \ x) in (ru \ x) is valid.

(NC13) the “Elements-of-Hull-not-Elements-of-ItSelf” Condition (EHnEIS-Cond):
forall y in e*(x): y non-in y ====> Normal(x) .
(NC14) the “Elements-of-Hull-not-Elements-of-its-Own-Hull” (EHnEOH-Cond):
forall y in e*(x): y non-in e^+(y) ===> Normal(x) .
(NC15) “Elements-of-Hull-not-Infinite-Descending-Element-Sequence” (EHnIDES-Cond):
forall y in e*(x): non-exist f: [Function(f) & f(0) = y & forall n in omega: f(n+1) in f(n)] ===> Normal(x) .
(NC16) “Elements-of-Hull-Founded” Condition (EH.Found-Cond):
forall y in e*(x): exist z in y: z intersection y = empty ===> Normal(x) .

(NBxyz) can be any arbitrary Normal Condition. Because we can also consider systems NAMix where “Normal” is constructed completely differently. (e.g. as already mentioned not depending on the elements y or the constructed set {x: A(x)}, but on the structure of the wff A, as in NF.) To investigate all these conditions is the program of NAM.[28]

(15) Extensions of NAM

Most systems NAMix described up to now are pure set theory. Of course we can always cover these pure set theory systems with a class theory (similar to NBG), getting the systems CNAMix of CNAM. That is important especially because sets and classes generated with the same wff A often have a different extension. (E.g., in some systems the universal set “us” is different from the universal class UC, which can contain abnormal sets like the Russell-set “ru” too (usually different from “us” in some systems), while “us” may not contain abnormal sets.) For classes over NAM-sets we use capital Latin letters X, Y, Z, … etc as variables and {|x: A(x)|} as class operator.

But NAM can also be extended into the other direction (i.e. downwards) if we restrict the set-constituting variable “x“ in the set operator to normal sets, i.e. use always {x: A(x) & Normal(x)}. For normal sets we introduce as new variables the small German letters german-x, german-y, german-z, … etc (abbreviated as g-x, g-y, g-z, … etc) and the normal-set-operator (NS-O) := {g-x: A(g-x)} or {/x: A(x)/}. We want to call this theory then the Germanized Naïve Axiomatic Mengenlehre GNAM and its systems GNAMix.[29] If a majority of the NAMix yields (after their Germanization) a unique same system GNAM! (of course up to equivalence), it would be an indication that this system GNAM! is an adequate representative of the formalization of Naïve Mengenlehre!

With the help of normal sets such systems (generated by normal conditions which allow counter-intuitive sets too) can also be made to work very well. We can define in such systems functions, relations, cardinals and ordinals, equality, subsets, power, etc, as precisely as those used in ZFC. If there is no danger of mixing it up we can drop the German letters and use, as accustomed, the Latin ones. And then everything mathematicians are doing in the Naïve Mengenlehre is as usual; only now they know (because of the experiments and the philosophical justification going hand in hand with them) that what they are doing is consistent, inspite of Gödel and the miscarriage of Hilbert’s program.

---

[28] Further Normal Conditions can be formulated and constructed with structural constraints on the formula A (like Stratified(A)). But we can also use completely different structural properties like “Supplement“ (= Counter-valisation, where all epsilons are replaced by non-epsilons in the expanded formula, and vice versa), Dualisation (where alle “and” are replaced by “or” and all general quantifiers by existence-quantifiers, and vice versa), and structural invariants of such constraints. It may also be possible to show with (AC) that some NC’s are equivalent, or we can derive for some of the NC’s an implication chain already in NAM0a or in other NAMix.

[29] In this case, ZFC would be for example GNAM0a without Eventuality Axioms. Of course (EFES) and (nNEM) would become redundant (or may have to be rejected), and (ElN-EA) deriveable for normal sets from (NE-Im) or (S-Im), because in GNAM the sets in the set operator are restricted to normal ones like the classes in the class operator are restricted to sets. Therefore we could also use GNAM0a as basis for ZFC. In both cases the missing ZF-axioms can be deduced from the Foundamental Axioms.

References:

Bernays, Paul & Fraenkel, Abraham
[1968] Axiomatic Set Theory. Springer Verlag, Heidelberg, New York.
Brunner, Norbert & Felgner, Ulrich
[2002] Gödels Universum der konstruktiblen Mengen. In: [Buldt & al, 2002].
Buldt, Bernd & Köhler, Eckehard & Stöltzner, Michael & Weibel, Peter & Klein, Carsten & DePauli, Werner
[2002] Kurt Gödel: Wahrheit und Beweisbarkeit, Band 2: Kompendium zu Gödels Werk. oebv&htp, Wien.
Casti, John & DePauli, Werner
[2000] Gödel: A Life of Logic. Perseus Publishing, Cambridge (MA).
DePauli-Schimanovich, Werner & Weibel, Peter
[1997] Kurt Gödel: Ein Mathematischer Mythos. Hölder-Pichler-Tempsky Verlag, Wien.
DePauli-Schimanovich, Werner: See also: Schimanovich [1971a] and [1971b].
[1998] Hegel und die Mengenlehre. In: Europolis3, Passagen-Verlag (Vienna, probably 2003).
Preprint at: http://www.univie.ac.at/bvi/europolis .
[2003?] The Notion of "Pathology" in Set Theory. Paper submitted to Journal of Philosophical Logic.
[2003?] Naïve Axiomatic Class Theory NACT: a Solution for the Antinomies of Naïve "Mengenlehre".
Paper intended for a journal of logic.
Davidson, Donald & Hintikka, Jaakko
[1969] Words and Objections. D. Reidel Publ. Comp., Dordrecht.
Feferman, Solomon & Dawson, John & Kleene, Stephen & Moore, Gregory & Solovay, Robert & Heijenoort, Jean van, [1990] Kurt Gödel: Collected Works, Volume II. Oxford University Press, New York & Oxford.
Felgner, Ulrich
[1985] Mengenlehre: Wege Mathematischer Grundlagenforschung. Wissensch. Buchgesellschaft, Darmstadt.
[2002] Zur Geschichte des Mengenbegriffs. In: [Buldt & al, 2002].
Forster, Thomas
[1995] Set Theory with an Universal Set. Exploring an Untyped Universe. (2nd Edition.)
Oxford Science Publ., Clarendon Press, Oxford.
Gödel, Kurt
[1938] The relative consistency of the axiom of choice and of the generalized continuum hypothesis.
In: [Feferman & al, 1990].
Goldstern, Martin & Judah, Haim
[1995] The Incompleteness Phenomenon. A. K. Peters Ltd., Wellesley (MA).
Goldstern, Martin
[1998] Set Theory with Complements. http://info.tuwien.ac.at/goldstern/papers/notes/zfpk.pdf
Halmos, Paul
[1960] Naive Set Theory. Van Nostrand Company Inc., Princeton (NJ).
Holmes, Randall
[1998] Elementary Set Theory with a Universal Set.
Vol. 10 of the Cahiers du Centre de Logique. Academia-Bruylant, Louvain-la-Neuve (Belgium).
[2002] The inconsistency of double-extension set theory.
http://math.boisestate.edu/~holmes/holmes/doubleextension.ps
Jech, Thomas
[1974] Procedings of the Symposium in Pure Mathematics (1970), Vol. XIII, Part 2, AMS, Provicene R.I. .
Jensen, Ronald Björn
[1969] On the consistency of a slight (?) modification of Quine's New Foundation.
In: [Davidson & Hintikka, 1969].
Köhler, Eckehart & Weibel, Peter & Stöltzner, Michael & Buldt, Bernd & Klein, Carsten & DePauli, Werner
[2002] Kurt Gödel: Wahrheit und Beweisbarkeit, Band 1: Dokumente und historische Analysen.
Hölder-Pichler-Tempsky Verlag, Wien.
Kolleritsch, Alfred & Waldorf, Günter
[1971] Manuskripte 33/'71 (Zeitschrift für Literatur und Kunst). Forum Stadtpark, A-8010 Graz, Austria.
Quine, Willard Van Orman
[1969] Set Theory and its Logic. Belknap Press of Harvard University Press, Cambridge (MA).
Rubin, Jean & Rubin, Herman
[1978] Equivalents of the Axiom of Choice. Springer, Heidelberg & New York
Schimanovich, Werner, [1971a] Extension der Mengenlehre. Dissertation an der Universität Wien.
[1971b] Der Mengenbildungs-Prozess. In: [Kolleritsch & Waldorf, 1971].
Schimanovich-Galidescu, Maria-Elena
[2002] Princeton – Wien, 1946 – 1966. Gödels Briefe an seine Mutter. In: [Köhler & al, 2002].
Scott, Dana, [1974] Axiomatizing Set Theory. In: Jech[1974] .
Suppes, Patrick, [1960] Axiomatic Set Theory. D. Van Nostrand Company Inc., Princeton (NJ).

On Non-associative Gröbner Bases

Lothar Gerritzen[1]

[1] *Bochum*

Abstract

In this article the basic notions of a theory of Gröbner basis for ideals in the non-associative, non-commutative algebra with unit freely generated by a set are discussed. The main result is a criterion for a system of polynomials to be a Gröbner basis. It can be seen as a non-associative version of the Buchberger criterion.

KEYWORDS: free magmas, planar binary rooted trees, non-associative free algebras, ideals, Gröbner bases, non-associative Buchberger criterion

1. Introduction

In the first volume on algebra of the Encyclopedia of Mathematical sciences published by the Academy of Sciences of the USSR in 1987, the author I.R. Shafarevich is also presenting his general views about basic notions of algebra. He has attemped a description of the place in mathematics occupied by algebra by drawing attention to the process of measuring for which H. Weyl has coined the word coordinatisation. This general idea is certainly of great importance because it challenges the Bourbaki view that algebra is a certain theory of structures of composition laws. However since the appearance of computer algebra this recent approach to describe algebra generally also seems to be insufficient.

Later in this treatise Lie algebras, Cayley numbers and other types of non-associative algebras are discussed. It is stated that no general theory of nonassociative algebras exists at present and the question is raised if perhaps such a theory is just not possible, [S], § 19, D. (p.201).

It seems to me that this scepticism derived from the principle of coordinatisation is no longer justified.

The non-associative, non-commutative algebra freely generated by one element x has a vectorspace basis given by the set of finite planar, binary rooted trees and it is nowadays completed clear that these objects are indispensable in computer science and other theoretical branches for abstract measurings, searchings and designs, see [Kn].

In this note I would like to show that there is an interesting algorithmic and computational theory of Gröbner bases for ideals in the non-associative non-commutative K-algebra $K\{X\}$ with unit element freely generated by a set X of variables over a field K. It can be considered as a branch of a general theory of non-associative algebras. In this context one should also mention the many publications on Hopf algebras of trees, see [CK], [LR], [BF] and recent work on operads, see [GK], or the non-associative exponential, logarithm and Hausdorff series, [DG], [GH].
A K-vectorspace basis for $K\{X\}$ is the magma $Mag'(X)$ with unit freely generated by X, see [B]. There are admissible orders $<$ on $Mag'(X)$ and for a non-zero polynomial f in $K\{X\}$ one has the leading term $f^<$ in $Mag'(X)$ of f.
A system G of generators for an ideal I in $K\{X\}$ is a Gröbner basis if the leading term $f^<$ of any non-zero f in I is a multiple of an element $g^<, g \in G$, in $Mag'(X)$.
The main result in this article is a criterion which guarantees that a given set of polynomials is a Gröbner basis. It is a non-associative version of the criterion of Buchberger, see section 6.
There are procedures similar to the Buchberger algorithm, [Bu], to construct non-associative Gröbner bases, see section 7.
In the sections 2 - 5 basic notions about free magmas, finite labelled planar binary trees and ideals in free magmas and free algebras are given. Proposition (4.3) is used in the proof of the Buchberger Criterion. In section 8 a few examples of Gröbner bases are discussed.

I thank the referees for valuable comments.

2. On free magmas and trees

A magma is a set N together with a binary operation on N which usually will be denoted by $._N$ or a dot.

This notion was introduced by Bourbaki, [B], Chap. 1, § 1. It has initialized the name of the Algebraic Programming Language Magma, see [CP], section (4.1), p. 46.
Let X be a set.

PROPOSITION 2.1: *There is a magma $Mag(X)$ with the following properties:*

(i) *X is a subset of $Mag(X)$*

(ii) *The multiplication on $Mag(X)$ is an injective map $\cdot : Mag(X) \times Mag(X) \to Mag(X)$ and the image of $\cdot$ is the complement of X in $Mag(X)$.*

$Mag(X)$ is uniquely determined by X up to isomorphisms and is called the magma freely generated by X.

Proof: The construction of $Mag(X)$ can be found in the literature, see for instance [B], [G1]. □

There is a unique morphism $deg : Mag(X) \to \mathbb{N}$, such that $deg(x) = 1$ for all $x \in X$ where $\mathbb{N}$ denotes the additive monoid of natural numbers. Then $deg(v)$ is called the degree of $v \in Mag(X)$.
Let T be a finite rooted tree, see [H], Chap. 15, p. 187. We denote by T^0 the set of nodes (vertices) of T, by $\bar{T}$ the set of edges of T and by w_T the root of T.
For any node a of T we denote by $val_T(a)$ the valence of T. It is the number of edges of T which are incident with a.

Definition: T is called binary if $val_T(w_T) \in \{0, 2\}$ and $val_T(a) \in \{1, 3\}$ for any node $a \neq w_T$ of T.

The nodes a of T with $val_T(a) \leq 1$ are called leaves (or end nodes) of T. $L(T)$ denotes the set of leaves of T. If there is a node a in T with $val_T(a) = 0$, then T consists of a single node.

Definition: A finite binary tree T together with a dissection $\bar{T} = \bar{T}^1 \dot{\cup} \bar{T}^2$ of the set of edges of T is called planar, if for any node a of T which is not a leaf of T there are edges $k_1 \in \bar{T}^{(1)}, k_2 \in \bar{T}^{(2)}$ incident with a and if $a \neq w_T$ then neither lies on the unique simple path from the rooted w_T to a. The edges in $\bar{T}^{(1)}$ (resp. $\bar{T}^{(2)}$) are called the left (resp. right) edges of T.

Definition: A finite planar binary rooted free T together with a map

$$\lambda : L(T) \to X$$

is called X-labelled.

Let T_1, T_2 be two finite planar binary rooted trees and φ a bijective map $T_1^0 \to T_2^0$. Then φ is called an isomorphism from T_1 onto T_2, if φ maps the root of T_1 onto the root of T_2 and if the map induced by φ on the system $Pot(T_1^0)$ of the subsets of T_1^0 maps $\bar{T}_1^{(i)}$ onto $\bar{T}_2^{(i)}$ for $i = 1, 2$. Then $\varphi(L(T_1)) = L(T_2)$. If λ_i is an X-labelling of T_i, then φ is an isomorphism from (T_i, λ_1) into (T_2, λ_2) if φ is an isomorphism of planar binary rooted trees and $(\lambda_2 \circ \varphi)|L(T_1) = \lambda_1$.
Let $PB(X)$ denote the set of all isomorphism classes of X-labelled finite planar binary rooted trees.
If $T, T' \in PB(X)$, then there is a unique $T \cdot T' \in PB(X)$ with the following properties: if the root of $T \cdot T'$ and the edges incident with it are removed from $T \cdot T'$, then the components of connectivity of this remaining graph consists of T and T'. This operation is called grafting and turns $PB(X)$ into a magma.
If $X = \{x\}$, then all labels of all elements in $PB(\{x\})$ are x and we can forget them. $PB(\{x\})$ is then called the magma of finite planar binary rooted trees and is also denoted by PB.
The following fact is fundamental for free magmas because it shows that there is an canonical combinatorial structure on the elements of $Mag(X)$. Strangely this view was not included in the presentations of free magmas by Bourbaki, [B], or

in the discussion by Kurosh, [K], of general free algebras. Maybe this was one reason that the general theory of non-associative algebras was not flourishing for some decades in the last century.
The proof of the following statement is simple, see also [R], p. 5 or [G2].

PROPOSITION 2.2: *(i) There is a unique morphism*

$$\eta : Mag(X) \to PB(X)$$

such that $\eta(x) =$ *tree consisting of a node only labelled with* x *whenever* $x \in X$.

(ii) η *is an isomorphism of magmas*

(iii) For any $v \in Mag(X), deg(v)$ *is equal to the number of leaves of* $\eta(v)$.

Proof: 1) Let $v \in Mag(X), n = deg(v)$. We define $\eta(v)$ by induction on n. If $n = 1$ then $v \in X$ and $\eta(v)$ is already defined. If $n > 1$, then $v = v_1 \cdot v_2, v_i \in Mag(X)$ and $deg(v_i) < n$. By induction hypothesis $\eta(v_i)$ is already defined and $\eta(v) := \eta(v_1) \cdot (v_2)$. Obviously η is a morphism, because there is only one decomposition of v into two factors.
2) Let $T \in PB(X)$ and let $l(T)$ denote the number of leaves of T. We want to show that η is surjective by induction on $l(T)$.
If $l(T) = 1$, then T has only one node and $T = \eta(x)$ for some $x \in X$. If $l(T) > 1$, then $T = T_1 \cdot T_2$ with $T_i \in PB(X)$ and $L(T) = L(T_1)\dot{c}upL(T_2)$ and thus $l(T) = l(T_1) + l(T_2)$.
By induction hypothesis there are $v_i \in Mag(X)$ such that $\eta(v_i) = T_i$. Then $\eta(v) = T$, if $v = v_1 \cdot v_2$.
3) using the argument in 2) one can also show that $deg(v) = l(\eta(v))$ for all v. This proves statement (iii).
4) Assume that η is not injective. Then there are $v, w \in Mag(X), v \neq w$, such that $\eta(v) = \eta(w)$.
We choose the pair v, w such that $n = deg(w)$ is minimal. Then $n = deg(v)$ by 3) and $n > 1$, because for $n = 1$ the tree $\eta(v)$ has only one node.
Let $v = v_1 \cdot v_2, w = w_1 \cdot w_2$ be decompositions in $Mag(X)$ and $\eta(v_i) = T_i, \eta(w_i) = S_i$. Then $T_1 \cdot T_2$ is isomorphic to $S_1 \cdot S_2$. An isomorphism between $T_1 \cdot T_2$ onto $S_1 \cdot S_2$ maps the root of $T_1 \cdot T_2$ to the root of $S_1 \cdot S_2$ and obviously also the subtree T_i of T to S_i. Thus it induces isomorphisms $T_i \to S_i$. It follows that $v_i = w_i$, as $deg(v_i) < n$. This is a contradiction to the assumption $v \neq w$. □

Let PB be the magma of finite planar binary rooted trees.
Let $T \in PB$ and $l(T) =: deg(T) = n$. The set $L(T)$ of leaves of T is a canonically ordered set. This ordering is defined by induction on n. This is trivial for $n = 1$. If $n > 1$ and $T = T_1 \cdot T_2$ we may assume that the set $L(T_i)$ of leaves of T_i is already ordered. This leads to an order on $L(T) = L(T_1)\dot{\cup}L(T_2)$ by define $b_1 < b_2$ for any $b_1 \in L(T_1), b_2 \in L(T_2)$.

Let now N be any magma and $v_1, ..., v_n \in N$.
We want to define $T(v_1, ..., v_n)$ by induction on n.
If $n = 1$, then it is v_1. If $n > 1$ and $T = T_1 \cdot T_2$ then

$$T(v_1, ..., v_n) = T_1(v_1, ..., v_{n_1}) \cdot T_2(v_{n_1+1}, ..., v_n)$$

if $n_1 = deg(T_1)$.
Informally $T(v_1, ..., v_n)$ is the product of $v_1, ..., v_n$ according to the bracketing induced by the tree $T \in PB$.

3. Admissible orderings on free magmas

A well-ordering $<$ on the free magma $M = Mag(X)$ is called admissible, if

(i) Whenever $a, b, c \in M$ and $a < b$, then $ac < bc$ and $ca < cb$.

(ii) Whenever $a, b \in M$, then $a < ab$ and $a < ba$.

We give the construction of an admissible ordering $<_{deg,1}$ on M which will be called the degree first factor ordering.
Fix a well-ordering on X.
Let $v, w \in M, v \neq w$.
Define $v <_{deg,1} w$, if $deg(v) < deg(w)$.
Assume now that $deg(v) = deg(w) = k$.
Define $v <_{deg,1} w$, if $v, w \in X$ and v is smaller than w relative to the fixed ordering on X.
Assume now that $v, w \notin X$. Then $k > 1$ and $v = v_1v_2, w = w_1w_2$ with $v_i, w_i \in M$ and $deg(v_i) < k, deg(w_i) < k$ for all i.
By induction on k we define $v <_{deg,1} w$, if $v_1 <_{deg,1} w_1$ or $v_1 = w_1$ and $v_2 <_{deg,1} w_2$.
Then $<_{deg,1}$ is a well-ordering because one can prove with induction on n that any subset of $\{v \in Mag(X) : deg(v) = n\}$ has a minimal element.

Assume that D is a subset of $Mag(X)$ of elements of degree n. We prove by induction on n that D has a minimal element relative to $<_{deg,1}$. If $n = 1$, it has a minimal element, because $D \subseteq X$.
If $n > 1$ and k is the minimal degree of the first factors of the elements in D, then $k < n$. Also the set D_k of all the first factors of elements from D of degree k has a minimal element v. Let $\bar{D}$ be the set of second factors of elements from D whose first factor is v. By induction hypothesis $\bar{D}$ has a minimal element w. Then $v \cdot w$ is minimal in D.

In a similar way one can define a degree second factor order $<_{deg,2}$ or $Mag(X)$.

Let $Mag'(X)$ be obtained from $Mag(X)$ by adjoining a neutral element which will be denoted by 1_X or 1. It will be called magma with unit freely generated by X. Any admissible order $<$ on $Mag(X)$ can be extended to $Mag'(X)$ by defining $1 < v$ for any $v \in Mag(X)$.

4. Ideals in free magmas

Let N be a magma, $a, b \in N$.

Definition: b is called a multiple of a in N, if there is a sequence

$$p = (c_0, ..., c_r), r \geq 0$$

such that $c_0 = a, c_r = b$, and $c_{i+1} = c_i \cdot d_i$ or $c_{i+1} = d_i \cdot c_i$ for all $0 \leq i < r$ with $d_i \in M$.

We call p also a path from a to b in N.
If b is a multiple of a in N, we also call a a divisor of b in N.

Let $I \subset N, I \neq \emptyset$.

Definition: I is called ideal in N of for all $a \in I, b \in N$, the elements $a \cdot b, b \cdot a \in I$.

It is easy to see that the set $< a >_N$ of all multiples of $a \in N$ in N is an ideal in N. It is the smallest ideal containing a and is also called the principal ideal generated by a.
The multiples of $v \in Mag'(X)$ in $Mag'(X)$ are the multiples of v in $Mag(X)$ if $v \neq 1$. If $v = 1$ then all elements in $Mag'(X)$ are multiples of v.

Proposition 4.1: *Let I be an ideal in $Mag'(X)$. There is a unique minimal set $\Omega \subset I$ such that*

$$I = \bigcup_{v \in \Omega} < v >_{Mag'(X)}$$

Ω is called the ideal basis of I.

Proof: If the neutral element 1_X of $Mag'(X)$ is contained in I, then $I =< 1_X >$.
Assume now that $1_X \notin I$.
Let

$$\Omega := I - (I \cdot Mag(X) \cup Mag(X) \cdot I)$$

It is the the union of $I \cap X$ with set of all

$$\omega = v \cdot w \in I$$

with $v, w \in Mag(X)$ and $v \notin I$ or $w \notin I$. I claim that $I = \cup_{\omega \in \Omega} < \omega >$
A obviously $< \omega > \subset I$ for all $\omega \in \Omega \subseteq I$.
Let now $t \in I, deg(t) = n$.
We show by induction on n, that $t \in < \omega >$ for some $\omega \in \Omega$.
If $n = 1$, then $t \in \Omega$. Consider now $n > 1$. If $t \notin \Omega$, there is a decomposition.

$$t = v \cdot w$$

with $v, w \in Mag(X)$ and $v \in I$ or $w \in I$.
Thus $v \in < \omega >$ or $w \in < \omega >$ as $deg(v) < n, deg(w) < n$ by the induction hypothesis. But then clearly $t \in < \omega >$. □

There is the following combinatorial description of the principal ideals in $Mag'(X)$. Let $v \in Mag'(X)$ and $T = \eta(v)$ be the tree in $PB(X)$ according to Prop. (2.2). Let a be a node of T and T_a the subtree of T whose nodes consist of all nodes b of T for which the simple path from the root of T to b passes through a.
If a is designed as root of T_a then T_a has a canonical structure of planar binary rooted tree and v_a is the unique element in $Mag(X)$ such that $\eta(v_a) = T_a$.

PROPOSITION 4.2: *The principal ideal in $Mag'(X)$ generated by an element $t \in Mag(X)$ consists of all $v \in Mag(X)$ for which there is a node a such that $v_a = t$.*

Proof: 1) Let a be a node of $S_1 \cdot S_2, S_i \in PB(X)$. Then a is a node of S_1 or of S_2 or it is the root of $S_1 \cdot S_2$. If it is a node of S_i, then $(S_1 \cdot S_2)_{\leq a} = (S_i)_{\leq a}$. If it is the root of $S_1 \cdot S_2$, then $(S_1 \cdot S_2)_{\leq a} = S_1 \cdot S_2$.
2) From 1) it follows easily that all the multiples S of $T \in PB(X)$ in $PB(X)$ have a node a with $S_{\leq a} = T$.
3) Let $T, S \in PB(X)$ and assume that there is a node a in S such that $S_{\leq a} = T$. We prove that S is a multiple of T by induction on $l(S)$. If a is the root of S, then $S_{\leq a} = S$ and $T = S$.
If a is not the root of S, then $S = S_1 \cdot S_2$ and a is a root of S_i for $i = 1$ or $i = 2$. Then $S_{\leq a} = (S_i)_{\leq a}$. As $l(S_i) < l(S)$ we may assume that S_i is a multiple of T. Then S is also a multiple of T. □

Example: Let $\Omega := \{x^n \cdot x^m : n, m \geq\} \subseteq Mag(\{x\})$. Then the magma ideal I generated by Ω in $Mag(\{x\})$ has Ω as basis, because if a is a node in the tree $\eta(x^n \cdot n^m) = T$, but not the root of T, then $T_{\leq a} = \eta(x^k)$ for some k. It follows that $x^n \cdot x^m$ is not a multiple of $x^{n'} \cdot x^{m'}$ if $(n', m') \neq (n, m)$. By Prop. (4.1) it follows that I has no finite set of generators.
The following result is used in the proof of Prop. 4.3.

LEMMA 4.1: *Let q_1, q_2 be nodes of $T \in B$ and assume that the subtrees $T_1 = T_{\leq q_1}, T_2 = T_{\leq q_2}$ of T have a common node.*
Then T_1, T_2 can also be considered as elements of PB and then T_1 is a divisor or a multiple of T_2 in PB.

Proof: Let v be a node in T which belongs to T_1 and T_2. Let P be the simple path in T from the root of T to v. It passes through q_1 and q_2. If it first passes through q_1, then $T_2 \subseteq T_1$ and T_2 is a divisor of T_1 in B. Otherwise $T_1 \subseteq T_2$ and T_1 is a divisor of T_2 in PB. □

PROPOSITION 4.3: *Let $w_1, ..., w_n \in Mag(X)$ and assume that w_i is not a multiple of w_j for all $i \neq j$.*
Let $< w_i >$ be the principal ideal in $Mag(X)$ generated by w_i and

$$I =< w_1 > \cap ... \cap < w_k > .$$

Then I is an ideal in $Mag(X)$ and $v \in Mag(X)$ is in I if and only if there is a tree $T \in PB$ and $u_1, ..., u_n \in X \cup \{w_1, ..., w_k\}$ of degree n such that

$$v = T(u_1, ..., u_n)$$

and such that for all i there is a j with $u_j = w_i$.

Proof: 1) If $v = T(u_1, ..., u_n)$ with the properties in the statement above, then $v \in < w_i >$ for all i by Prop. (4.1). Thus $v \in I$.
2) Let $v \in I$ and Q be the set of nodes q of the tree $S = \eta(v) \in PB(X)$ for which the subtree $S_{\leq q_1}$ is contained in $\{\eta(w_1), ..., \eta(w_k)\}$. Then $S_{\leq q'} \cap S_{\leq q} =$ empty for $q \neq q'$, see Lemma (4.1).
If we remove all non-root nodes of $S_{\leq q}$ for all q, we get a tree $T \in PB$.
If the i-th leaf of T is a leaf of S, then we put $v_i =$ label of S at this leaf. If it is $q \in N$, then $v_i := w_j$, if $\eta(w_j) = S_{\leq q}$.
Then $T(v_1, ..., v_n) = S$ and for any i there is j such that $w_i = v_j$. □

5. Gröbner bases of ideals in free algebras

Denote by $K\{X\}$ the magma algebra of $Mag'(X)$ over a field K. This object was studied by Kurosh in [K] and also occurs in the work of Lazard, [L]. A K-vectorspace base of $K\{X\}$ is $Mag'(X)$.

Let I be a K-subvectorspace of $K\{X\}$

Definition: I is called ideal of $K\{X\}$ if $f \cdot g, g \cdot f \in I$ whenever $f \in I$ and $g \in K\{X\}$.

Let $F \subset K\{X\}$. There is a smallest ideal $I(F)$ in $K\{X\}$ which contains F.

Let $f, g \in K\{X\}$.

Definition: g is called a multiple of f in $K\{X\}$, if there is a sequence $p = (f_0, f_1, ..., f_r), r \geq 0$, such that

(i) $f_0 = f, f_r = g, f_i \in K\{X\}$.

(ii) $f_i = h_i \cdot f_{i-1}$ or $f_i = f_{i-1} \cdot h_i$ with $h_i \in K\{X\}$.

Proposition 5.1: *$I(F) = K$-vectorspace generated by $\bigcup_{f \in F} \langle f \rangle$ where $< f >$ denotes the set of all multiples of f in $K\{X\}$.*

Let $<$ be an admissible order on $Mag(X)$ and $f \in K\{X\}, f \neq 0$.
We denote the leading term of f relative to this order by $f^<$ and consider $f^<$ as element in $Mag(X)$. The leading coefficient of f is denoted by $c(f) \in K$.
Then the leading term of $f - c(f) \cdot f^<$ is less than $f^<$.
Let now I be an ideal in $K\{X\}, I \neq \{0\}$ and $F \subset I, 0 \notin F$. Then $I^< := \{f^< : f \in I, f \neq 0\}$ is a magma ideal in $Mag(X)$.
At this point we do not assume F to be finite, because ideals in $K\{X\}$ are not always finitely generated even if $\sharp X = 1$, see Example 2 in section 8 or the Example after Proposition 4.2.

Definition: F is called Gröbner bases of I, if the ideal $I^<$ is generated by $F^< := \{f^< : f \in F\}$.

Assume now for simplicity that $c(f) = 1$ for all $f \in F$.
Now we introduce a relations $\mapsto_F$ on $K\{X\}$ which will be called the relation of elementary reductions relative to F.
Let $h \in K\{X\}, h \neq 0$, and let w be a term in h whose coefficient $c_w(f) \neq 0$.
Assume that $f \in F$ and that the leading term $f^<$ of f is a factor of w in $Mag(X)$.
Let $p = (v_0, v_1, ..., v_r)$ be the path from $v_0 = v$ to $v_r = f^<$.
We construct a sequence $(f_0, ..., f_r)$ as follows: $f_0 := f$ and if f_{i-1} is already defined and if $v_i = v_{i-1} \cdot u_i$ with $u_i \in Mag(X)$, then $f_i := f_{i-1} \cdot u_i$.
If however $v_i = u_i \cdot v_{i-1}$ with $u_i \in Mag(X)$, then $f_i := u_i \cdot f_{i-1}$.
Then w is the leading term of f_r and the coefficient of f_r relative to the term w is equal to 1.
Let $h' := h - c_w(h) \cdot f_r$. Then the coefficient of h' relative to the term w is zero.
Define $h \mapsto_F h'$, if h' is obtained from h in the procedure constructed above.
Let $\mapsto_F^*$ be the reflexive and transitive closure of $\mapsto_F$ and $N(F)$ be the space of all $h \in K\{X\}$ for which no term of h with coefficient $\neq 0$ is a multiple of any $f^<, f \in F$.
$N(F)$ is a vectorspace. It is called the space of normal polynomials with respect to F.

PROPOSITION 5.2: *For any $h \in K\{X\}$, $h \neq 0$, there is $h' \in N(F)$ such that $h \mapsto_F^* h'$.*
We call h' a normal form of h with respect to F. It is not uniquely defined by h in general.

The importance of Gröbner bases can be seen in the following result, which can be called the Macauly decomposition. Its proof is completely analog to the commutative and non-commutative case, see[BW], [M], [MR].

COROLLARY 5.1: *(i) $N(F) \oplus I(F) = K\{X\}$, if F is a Gröbner bases of $I(F)$*

(ii) The restriction of the residue class homomorphism $K\{X\} \to K\{X\}/I(F)$ onto $N(F)$ is bijective.

Definition: A Gröbner basis G of an ideal I in $K\{X\}$ is called reduced, if

(i) all polynomials $g \in G$ are unitary

(ii) $g_1^< \neq g_2^<$ for $g_1, g_2 \in G, g_1 \neq g_2$

(iii) $\{g^< : g \in G\}$ is the basis of the magma ideal $I^<$.

(iv) if t is a term occuring in $g - g^<$, then $t \notin I^<$.

It is easy to check that any ideal has a unique reduced Gröbner bases.
Reduced non-associative Gröbner bases have appeared in a formula for the Hilbert series of graded algebras, see [G2]. It expresses the Hilbert series of any graded algebra of finite type through the series

$$q(\zeta) = \frac{1}{2}(1 - \sqrt{1 - 4\zeta})$$

and the generating series of a reduced Gröbner bases for the relation ideal with respect to a system X of homogeneous algebra generators. More precisely it has the form

$$1 + q(G_X(\zeta) - G_\Gamma(\zeta))$$

where G_X is the generating series of X and G_Γ is the generating series of a minimal Gröbner basis of the ideal of relation in $K\{X\}$ with respect to degree, see [G2], Proposition (3.1).

Remark: The theory of Gröbner bases should also be presented in free algebras over rings, for example over Zacharias rings. This can be done by using methods from [GTZ]

6. The criterion of Buchberger

Let $F \subseteq K\{X\}$ and assume that all $f \in F$ are unitary, i.e. $c(f) = 1$.
Let $f, g \in F, f \neq g$.

We want to define the s-polynomial $spol(f, g)$ of f and g.
Case 1: $f^<$ is a factor of $g^<$.
Let $p = (v_0, ..., v_r)$ be the path between $f^<$ and $g^<$.
Then a sequence $(s_0, ..., s_r)$ is defined by putting $s_0 = f$. It s_{i-1} is already defined, then

$$s_i = u_i s_{i-1} \text{ if } v_i = u_i v_{i-1}$$

$$s_i = s_{i-1} u_i \text{ if } v_i = v_{i-1} u_i$$

Then

$$spol(f, g) := s_r - g$$

Case 2: $g^<$ is a factor of $f^<$.
Then $spol(f, g) := -spol(g, f)$.
Case 3: $g^<$ is not a factor and not a multiple of $f^<$. Then $spol(f, g) := 0$.

The following statement is the generalization of the classical criterion of Buchberger.
It is simple because there are no overlaps as in the non-commutative case.

PROPOSITION 6.1: *F is a Gröbner basis of $I = I(F)$, if $spol(f,g) \mapsto_F^* 0$ for all $f, g \in F, f \neq g$.*

Proof: 1) Let $f_1, ..., f_r \in K\{X\}, v_i = f_i^< \in Mag(X)$ and assume that no v_i is a proper multiple of v_j in $Mag(X)$ for all i and j. Let $h = h_1 + ... + h_r, h_i \in < f_i >:=$ ideal in $K\{X\}$ generated by f_i and $h_i \neq 0$ for all i.
Let $m := max_{i=1}^r(h_i^<)$. We may assume that $h_i^< = m$ for $1 \leq i \leq r'$ and $h_i^< < m$ for $r' < i \leq r$. Then $r' \geq 2$.
From Proposition (4.3) we get a finite, planar binary rooted tree $T \in PB$ of degree n and $u_1, ..., u_n \in \{v_1, ..., v_r\} \cup (X)$ such that

$$m = T(u_1, ..., u_n)$$

and for each $i \leq r'$ there is an index j such that $u_j = v_i$.
Let now $\tilde{u}_i := f_j$ if $u_i = v_j$ and let $\tilde{u}_i = u_i$, if $u_i \notin \{v_1, ..., v_r\}$. Then $\tilde{m} := T(\tilde{u}_1, ..., \tilde{u}_n) \in K\{X\}$ and $\tilde{m} \in < f_i >$ for $i \leq r'$.
For any $i, 1 \leq i \leq r'$, there is $\lambda_i \in K$ such that $c(h_i) = \lambda_i$. We put $\lambda_i = 0$ for $i > r'$ and let $\tilde{h}_i := h_i - \lambda_i \tilde{m}$.
Then $\tilde{h}_i \in < f_i >$ for all i because one can check that $h_i^< \in < v_i >$ for $1 \leq i \leq r'$.
Assume now that $h^< < m$. Then $\sum_{i=1}^{r'} \lambda_i = 0$ and $\sum_{i=1}^{r} \tilde{h}_i = h$.
It is easy to check that $\tilde{h}_i^< < m$ for all i.
By using this construction several times we obtain a decomposition

$$h = h_1' + ... + h_r'$$

with $h_i' \in < f_i >$ and $(h_i')^< = h^<$ for some $i, (h_j')^< \leq h^<$ for all j.

2) Let $G := \{g \in F :$ no proper factor of g in $Mag(X)$ is contained in $F^<\}$. Using the construction in 1) we can prove that G is a Gröbner basis of the ideal $I(G)$ generated by G. By induction on $deg(f)$ and the fact that f can be reduced relative to G to zero one can show that $F \subseteq I(G)$.
Thus F is Gröbner basis of $I(F) = I(G)$. □

COROLLARY 6.1: *Let $f_1, ..., f_r \in K\{X\}, f_i \neq 0, f_i \in Mag(X)$ and assume that $f_i^<$ is not a divisor of $f_j^<$ for $i \neq j$ in $Mag(X)$. Then $\{f_1, ..., f_r\}$ is a Gröbner basis of the ideal in $K\{X\}$ generated by $f_1, ..., f_r$.*

Remark: It seems to be of interest to extend the notion of Gröbner bases to more general objects in universal algebra, see [BS], Chap. II, §1, §10.
A case of particular importance seems to be the algebra whose system of monomials is the set P of all planar rooted trees, which are not necessarily binary. The grafting defines n-ary operations $P^n \to P$ for all $n \geq 1$.
The automorphism group of the power series version of this algebra is the dual to the Hopf algebra considered in [CK] for applications in quantum field theory.

7. Algorithm

Let $F = \{f_1, ..., f_n\}$ be a system of polynomials in $K\{X\}, f_i \neq 0$, and $I = I(F)$ the ideal generated by F.
We want to sketch the quite simple algorithm to compute a Gröbner basis for I.
If $v_i = f_i^<$ and v_i is not a multiple of v_j for $i \neq j$, in $Mag(X)$, then F is a Gröbner basis of I by Corollary (6.1).
Otherwise we choose a pair $(i, j), i \neq j$, such that v_i is a divisor of v_j.
Let f_j' be a normal form of f_j relative to $\{f_i\}$. Then the leading term of f_j' is smaller than v_j.
Let $F' := \{f_1', ..., f_n'\}$ with $f_i' := f_i$ for $i \neq j$.
Then $I(F) = I(F')$. We repeat the procedure just described for F now with F'.
Thus we get a sequence $F^{(k)} = \{f_1^{(k)}, ..., f_n^{(k)}\}, n \geq 1$, with $(F^{(k)})' = F^{(k+1)}$ as long as $F^{(k)}$ does not satisfy the assumption of Corollary (6.1).
This sequence cannot be infinite. If it would be infinite, there would exist an index i such that the leading term $v_i^{(k)}$ of $f_I^{(k)}$ would not be constant as function of k. As it is decreasing, this is a contradiction to the well-ordering property of $<$.

Remark: The procedure just described can sometimes also be applied if F is infinite in cases where it is given by "regular" expressions.
It leads to theoretical results as in section 8, Exercise 2.

8. Examples

Example 1: Three-dimensional algebras

Let $f_1 = x^3 - \alpha, f_2 =^3 x - \beta, f_3 = x^2 \cdot x^2 - \gamma$ with $\alpha = \alpha_0 + \alpha_1 x + \alpha_2 x^2, \beta = \beta_0 + \beta_1 x + \beta_2 x^2, \gamma = \gamma_0 + \gamma_1 x + \gamma_2 x^2 \in K\{X\}$ where $x^3 = x \cdot x^2, ^3\, x = x \cdot x^2$.
Then f_1, f_2, f_3 is a Gröbner basis of the ideal generated by f_i, f_2, f_3 in $K\{X\}$ and $A = K\{X\}/I$ is a three-dimensional K-algebra with K-base $1, \bar{x}, \bar{x}^2$.
The left multiplication by $a = a_0 + a_1\bar{x} + a_2\bar{x}^2$ is given by the matrix

$$m_a = \begin{pmatrix} a_0 & a_2\beta_0 & a_1\alpha_0 + a_2\gamma_0 \\ a_1 & a_0 + a_2\beta_1 & a_1\alpha_1 + a_2\gamma_1 \\ a_2 & a_1 + a_2\beta_2 & a_0 + a_1\alpha_2 + a_2\gamma_2 \end{pmatrix}$$

and $det(m_a)$ is a homogeneous polynomial of degree 3 in the variables a_0, a_1, a_2.
If the set of zeros in the projective plane of this form has no K-rational point, then A is a division algebra.
It seems interesting to determine all $\alpha, \beta, \gamma \in \mathbb{Q}\{x\}$ for which A is a division algebra.

Example 2: Non-associative relations for the free associative algebra
Let $Mon(X)$ be the monoid freely generated by X. There is a canonical morphism $\eta : Mag(X) \to Mon(X)$ such that $\eta(x) = x$ for all $x \in X$. In [R],

(4.1), $\eta(w)$ is called the foliage of $w \in Mag(X)$. There is a unique morphism $deg : Mon(X) \to (\mathbb{N}, +)$ such that $deg(1) = 0, deg(x) = d(x)$ for all $x \in X$.
The associative K-algebra $K\langle X\rangle$ with unit freely generated by X is the monoid algebra $K[Mon(X)]$ of $Mon(X)$ over K.
The morphism $\eta : Mag(X) \to Mon(X)$ extends to a K-algebra homomorphism $\bar{\eta} : K\{X\} \to K\langle X\rangle$. If $J_a = ker(\bar{\eta})$ then J_a is a ideal in $K\{X\}$ and $K\langle X\rangle \cong K\{X\}/J_a$.
Let now $<$ be the degree first factor ordering constructed in section 3.
Let $w = x_1x_2...x_n \in Mon(X)$ be a word of lenght $n \geq 1, x_i \in X$ for all i. Define $c(w) = x_1$ if $n = 1$ and $c(w) := x_1 \cdot c(x_2 \cdot ... \cdot x_n) \in Mag(X)$. It is easy to check that $c(w) \leq v$ for all $v \in Mag(X)$ with $\eta(v) = w$.
Let $r \geq 3$ and $\Gamma_r := \{c(v)c(w) - c(vw) : v$ is a word of lenght $n \geq 2, n < r$ and w is a word of lenght $r - n$ over $X\}$. One can show

PROPOSITION 8.1: $\Gamma := \cup_{r=3}^{\infty}\Gamma_r$ *Gröbner basis of* J_a.

Proof: 1) For any $v \in Mag(X)$ let $|v|$ be the underlying planar binary tree in $Mag(\{\zeta\})$ of v.
Then $g_v := v - c(\eta(v)) = 0$ if and only if $|v|$ is a right comb ζ^n defined by $\zeta^1 = \zeta$ and $\zeta^n := \zeta \cdot \zeta^{n-1}$ for $n > 1$. Also $c(\eta(u)) = |v|(u_1, ..., u_n)$ if $\eta(v) = u_1u_2, ...u_n \in Mon(X)$ with $u_i \in X$.
Then $g_v^< = v$, if $g_v \neq 0$.
It is easy to check that the system

$$\{g_v : v \in Mag(X), g_v \neq 0\}$$

is a K-basis of J_a.
Also $J_a^< = \{v \in Mag(X) : |v|$ is not a right comb$\}$. $J_a^<$ is a magma ideal in $Mag(X)$ because $|v_1 \cdot v_2| = |v_1| \cdot |v_2|$ and $|v_1 \cdot v_2|$ is a right comb only if $|v_1| = \zeta$ and $|v_2| = \zeta^{n-1}$ for some n.

2) Let Ω be the minimal set of generators of $J_a^<$.
We show that $\Omega = \Gamma$.
Let $t \in J_a^<, t = t_1 \cdot t_2$. If $t \in \Omega$, then both $t_1, t_2 \notin J_a^<$ and $|t_1|, |t_2|$ are both right combs in $Mag(\{\zeta\})$. In this case $t_i = c(\eta(t_i))$ and $t \in \Gamma$.
If $t \in \Gamma$, then obviously $t = t_1 \cdot t_2$ with $t_1, t_2 \notin J_a^<$ and $|t_1|, |t_2|$ are right combs.
If t is a proper multiple of some $v \in J_a^<$, then $|v|$ is a factor of $|t_1|$ or $|t_2|$ and is thus also a right comb.
This is a contradiction which shows that $t \in \Omega$. □

Example 3: Free alternative algebras
Let $I_{alt}(X)$ be the ideal in $K\{X\}$ generated by the following system of elements:
$f^2g - f(fg), (fg)f - f(gf), (fg)g - fg^2$
for all $f, g, h \in K\{X\}$. The algebra $Alt(X) := K\{X\}/I_{alt}(X)$ is called the alternative algebra freely generated by X.

A theorem of Artin, see [KS], (2.3), p. 224, states that $Alt(x), Alt(\{x, y\})$ are associative, but $Alt(X)$ is not associative if $\sharp X \geq 3$.
We raise the question to determine a Gröbner basis for $I_{alt}(X)$.
The algebra of Cayley numbers is alternative and generated by three elements i, j, l. What is the reduced Gröbner basis of the ideal of relations?

References

[B] *Bourbaki, N.*: Algebra, Chap. I - Chap. X, Hermann, Paris, 1971

[Bu] *Buchberger, B.*: An Algorithm for Finding a Basis for the Residue Class Ring of a Zero-Dimensional Polynomial Ideal (German), PhD Thesis, University of Innsbruck, Institute for Mathematics, 1965

[BF] *Brouder, Ch. - Frabetti, A.*: QED Hopf algebras on planar binary trees, Preprint, Universitäten Paris / Lyon, December 2001

[BS] *Burris, S. - Sankappanavar, H. P.*: A course in Universal Algebra, http:// www.thoralf.uwaterloo.ca/htdocs/ualg.html

[BW] *Becker, T. - Weispfennig, V.*: Gröbner Bases: A Computational Approach to Commutative Algebra, Springer-Verlag, New York, 1993

[CK] *Connes, A. - Kreimer, D.*: Renormalization in quantum field theory and the Riemann-Hilbert problem I: the Hopf algebra structure of graphs and the main theorem, Comm. Math. Phys. 210 n.1, 249-273, 2000

[CP] *Cannon, J. J. - Playoust, C.*: An Introduction to Algebraic Programming with Magma, University of Sydney, 1997

[DG] *Drensky, V. - Gerritzen, L.*: Non-associative exponential and logarithm, Manuskript 2002

[G1] *Gerritzen, L.*: Grundbegriffe der Algebra, Vieweg, 1994

[G2] *Gerritzen, L.*: Hilbert series and non-associative Gröbner bases, manuscripta math. 103, 161-167, Springer-Verlag, Heidelberg, 2000

[GH] *Gerritzen, L. - Holtkamp, R.*: Co-Addition for Free Non-Associative Algebras and The Hausdorff Series, Manuskript 2002

[GK] *Ginzburg, V. A. - Kapranov, M. M.* : Koszul duality for operads, Duke Math. J. 76, 1994

[GTZ] *Gianni, P. - Trager, B. - Zacharias, G.* : Gröbner bases and primary decomposition of polynomial ideals, J. Symbolic Comput. 6, no. 2-3, p. 149-167, 1988

[H] *Harary, F.*: Graph Theory, Addison-Wesley, London, 1968

[K] *Kurosh, A.*: Non-associative free algebra and free products of algebra, Res. Math. (Mat. Sbornik) N.S. 20(62), (1947), 238-262

[Kn] *Knuth, D.E.*: The Art of Computer Programming, Vol. 1, Fundamental Algorithmus, Reading, Addison-Wesley, 1973

[KS] *Kuzmin, E.-N. - Shestakov, I.P.* : Non-Associative Structures, in: Encyclopedia of Mathematical Sciences, Vol. 57, Algebra VI (Kostrikin - Shafarevich Eds.), Vol. 57, Springer-Verlag, 1995

[L] *Lazard, M.*: Lois de groupes et analyseurs, Ann. Ecole Norm. Sup. Paris, 72 (1955)

[M] *Mora, T.*: An introduction to commutative and non-commutative Groebner baes, Theor. Comp. Sci. 134, p. 131-173 (1994)

[MR] *Madlener, K. - Reinert, B.*: Computing Gröbner bases in monoid and groups rings, Proc. ISSAC '93, p. 254-263, ACM (1993)

[LR] *Loday, J.L.- Ronco, M.O.*: Hopf algebra of the planar binary trees, Adv. Math. 139, 293-309, 1998

[R] *Reutenauer, C.*: Free Lie Algebras, Oxford Univ. Press, 1993

[S] *Shafarevich, I.R.*: Basic Notions of Algebra, in: Encyclopaedia of Mathematical Sciences, Vol. 11, Algebra I, (Kostrikin, A.I. - Shafarevich Eds.), Springer-Verlag, 1990

Address: Prof. Dr. Lothar Gerritzen

Ruhr-Universität Bochum
Fakultät für Mathematik
D 44780 Bochum
Germany

Incremental Decoding *

P. Gianni[1] and B. Trager[2]

[1] *Dipartimento di Matematica, Via Buonarroti 2, 56127 Pisa, ITALY*

[2] *IBM T.J.Watson Research Center, 1101 Kitchawan Road, Route 134, Yorktown Heights, NY 10598, USA*

Abstract

In this paper we examine a group of decoding algorithms which can be characterised as being incremental, in that they update the solution to a previous decoding problem to produce the solution to a new one. Our main contribution is to show that all of these algorithms can be unified under the notion of adding a linear constraint to a solution module. Gröbner bases are used not as a technique for constructing such bases, but as a framework for proving properties about them.

KEYWORDS: Error Correcting Codes, Groebner Bases, Reed-Solomon codes, Erasure Decoding, Burst Decoding, Free Modules

1. Introduction

The decoding process for alternant error correcting codes (which include Reed-Solomon and BCH codes as special cases) is reduced to the resolution of a congruence equation, called the key equation, which is of the form $p \equiv qs (\mathrm{mod} x^t)$, where $p, q, s \in F[x]$, $t \in \mathbf{N}$, and F is a finite field. Using syndromes computed from the received block of symbols, we get the syndrome polynomial s, while p and q represent the error evaluator and error locator polynomial, respectively. The desired solutions (p, q) must satisfy the degree constraint $\deg p < \deg q$.
The term erasures indicates potential error positions whose locations are known, represented as roots of a polynomial ψ, and in this case we require the solutions (p, q) to satisfy $\psi | q$.
In both cases we want that $\deg q$ is minimal among all the solutions. As already remarked by some authors (1),(2), this problem can be rephrased in terms of Gröbner bases of submodules of $A^2 = F[x] \times F[x]$ with respect to a suitable

*This research was partially performed with the contribution of M.U.R.S.T.

module term order such that the desired solution appears in a Gröbner basis for the module.

In this paper we examine a group of decoding algorithms which can be characterised as being incremental, in that they update the solution to a previous decoding problem to produce the solution to a new one. While most of these basic algorithms have appeared before (5), (4), they were each accompanied by a long, complicated ad-hoc proof of correctness. Our main contribution is to show that all of these algorithms can be unified under the notion of adding a linear constraint to a solution module. Gröbner bases are used not as a technique for constructing such bases, but as a framework for proving properties about them. Thus we use the theory of Gröbner bases of modules to prove the correctness of our constructions. At the end of the paper we include an application to burst error correction, which shows how our incremental algorithms can be combined to yield a new efficient solution to an important practical problem in coding theory.

2. Linear Constraints

Let F be a finite field, $A = F[x]$ and $A^2 = A \times A$ the free module of rank 2 over A. Let T be the set of terms in A^2, i.e. the set of elements of the form $\{(x^i, 0), (0, x^j) \mid i, j \in \mathbf{N}\}$, so that every element $m \in A^2$ can be written in a unique way as a linear combination of terms with coefficients in F. We define a total ordering $<$ on the set of terms T as follows:

Definition: $(a_1, b_1) < (a_2, b_2) \iff$

- $\deg a_1 < \deg a_2$ if $b_1 = b_2 = 0$
- $\deg b_1 < \deg b_2$ if $a_1 = a_2 = 0$
- $\deg a_1 < \deg b_2$ if $b_1 = a_2 = 0$

Given any element $m \in A^2$, we define $lt(m)$ as the largest term appearing in its representation, and naturally extend the ordering from T to A^2. For $m \in A^2$, if $lt(m) = (x^p, 0)$ (resp. $(0, x^q)$) we say that m has its leading term on the left (resp. on the right). We will say that two elements $m_1, m_2 \in A^2$ have leading terms on opposite sides if one has its leading term on the left and the other has its leading term on the right. This property is used to characterize a Gröbner basis for a submodule $M \subset A^2$.

It is well known (6) that if $M \subset A^2$ is a submodule, then M is a free-module of rank ≤ 2. In the rest of the paper we will restrict our considerations to submodules of rank equal to 2.

Proposition 2.1: *Let M be a submodule of $(A^2, <)$ then we have:*

- *Any pair of elements of M with leading terms on opposite sides and of minimal degrees forms a Gröbner basis.*

- *Any basis of M whose leading terms are on opposite sides is a Gröbner basis.*

Remark: We define minimal element in M, any element whose leading term is a minimal element in the set of all the leading terms of elements of M, $Lt(M)$. Any such element is unique up to scalar multiple.

In all the applications we have in mind, we will describe a module containing all solutions to a given equation, and we are interested in elements (a, b) such that $\deg a < \deg b$ and $\deg b$ minimal. Under our choice of ordering, such a pair will have its leading term on the right. A reduced Gröbner basis for the module will have exactly one element with leading term on the right and will have minimal degree among such elements. Thus we are guaranteed to find the sought for minimal solution in a Gröbner basis. The question of uniqueness is addressed in the following proposition:

PROPOSITION 2.2: *If $\{(a_L, b_L), (a_R, b_R)\}$ are a reduced Gröbner basis for a module with (a_L, b_L) having leading term on the left and (a_R, b_R) having leading term on the right, then:*

- $\deg b_R$ *is minimal among all elements having leading term on the right.*
- $(a_R, b_R) < (a_L, b_L) \iff \deg b_R \leq \deg a_L$. *In this case all minimal elements on the right are of the form $c(a_R, b_R)$ for $c \in F$.*
- $(a_R, b_R) > (a_L, b_L) \iff \deg b_R > \deg a_L$. *In this case if we let $d = \deg b_r - \deg a_L$, then all minimal elements on the right can be expressed as $c(a_R, b_R) + p(a_L, b_L)$ where $c \in F$ and $p \in A \mid \deg p < d$.*

Thus we see that in the non-unique case we can parametrize the space of minimal solutions. This can be exploited for list decoding.

From proposition 1 it follows that if we want to compute a Gröbner basis for M and we are given a set of generators, we don't need the complete version of Buchberger's algorithm, but only to reduce the generators until we find those of minimal degrees.

We will need the concept of the *discriminant* of a module M. As said before we will restrict ourselves to the case of submodules of A^2 of rank 2.

If $M \subset A^2$ is a submodule of rank 2 and $\mathcal{B} = \{m_1, m_2\}$ is a basis for M, we can express m_1 and m_2 in terms of a basis $\mathcal{E} = \{e_1, e_2\}$ of A^2. We can assume that $\mathcal{E}$ is the canonical basis $e_1 = (1, 0)$ and $e_2 = (0, 1)$. In this way

$$m_1 = b_{11}e_1 + b_{21}e_2$$

$$m_2 = b_{12}e_1 + b_{22}e_2.$$

The 2×2 matrix $B = (b_{ij})$ is called the *relation matrix* of the basis $\mathcal{B}$ in terms of the basis $\mathcal{E}$, when $\mathcal{E}$ is the canonical basis of A^2 we will omit the reference to $\mathcal{E}$.

Definition: Given a submodule M of A^2 with basis $\mathcal{B} = \{m_1, m_2\}$, the discriminant of $\mathcal{B}$, denoted by $discr(\mathcal{B})$, is the element of A given by the determinant of the relation matrix B of the basis $\mathcal{B}$. Moreover we define the discriminant of M the ideal of A generated by $discr(\mathcal{B})$.

Remark that the discriminant of M is independent of the chosen basis. We will use the following properties of the discriminant:

PROPOSITION 2.3: *Let $M \subseteq N$ be submodules of rank 2 of A^2, then*

(i) $discr(N)|discr(M)$

(ii) $M = N$ *if and only if* $discr(M) = discr(N)$

The following definitions and properties are the basis for our construction.

Definition: Let $M \subset A^2$ be a submodule of rank 2, $\alpha \in F$. Any A-linear map from M to A followed by evaluation at α will be called a linear constraint at α, L_α. We define $\widetilde{M_{L_\alpha}} = \{m \in M | L(m) = 0\} = \mathrm{Ker}(L_\alpha)$.

Remark: $\widetilde{M_{L_\alpha}}$ is a submodule of M which is free of rank 2 since it contains $(x - \alpha)M$.

We will show how to construct a set of generators and then a Gröbner basis for $\widetilde{M_{L_\alpha}}$ from a Gröbner basis for M.

PROPOSITION 2.4: *Let $\widetilde{M_{L_\alpha}} \subset M$ be the submodule of M defined by a linear constraint L_α, $\alpha \in F$, assume that $L_\alpha(M) \neq 0$ and let $B = \{m_1, m_2\}$ be a basis for M. Then if we define: $b_i = (x - \alpha)m_i$, $i = 1, 2$ and $b_3 = L_\alpha(m_2)m_1 - L_\alpha(m_1)m_2$ the set $\{b_1, b_2, b_3\}$ is a set of generators for $\widetilde{M_{L_\alpha}}$.*

Proof: Consider $m \in \widetilde{M_{L_\alpha}}$, there exists $d_1, d_2 \in A$ such that $m = d_1 m_1 + d_2 m_2$. It is possible to find $s_i \in A$ and $r_i \in F$ such that $d_i = s_i(x - \alpha) - r_i$ for $i = 1, 2$. Hence $n = m - (s_1 b_1 + s_2 b_2) = r_1 m_1 + r_2 m_2 \in \widetilde{M_{L_\alpha}}$. From the assumption that $L_\alpha(m) = 0$ and the definition of b_3 it follows that there exists $k \in F$ such that $n = k b_3$. □

PROPOSITION 2.5: *Let $M \subset A^2$ be a submodule of rank 2, let $B = \{m_1, m_2\}$ with $m_1 < m_2$ be a Gröbner basis of M with respect to $<$, let L_α be a linear constraint and $\widetilde{M_{L_\alpha}} = Ker(L_\alpha)$. Let*

$$\{b_1 = (x - \alpha)m_1$$
$$b_2 = (x - \alpha)m_2$$
$$b_3 = L_\alpha(m_2)m_1 - L_\alpha(m_1)m_2\}$$

be the set of generators for $\widetilde{M_{L_\alpha}}$ as given in the previous proposition. Define $n_2 = b_3$ and n_1 as follows:

- *If* $L_\alpha(m_1) \neq 0$ *then* $n_1 = b_1$
- *If* $L_\alpha(m_1) = 0$ *then* $n_1 = b_2$

then $\{n_1, n_2\}$ *is a Gröbner basis for* $\widetilde{M_{L_\alpha}}$.

Proof: Since $\{m_1, m_2\}$ is a Gröbner basis for M, m_1 and m_2 have leading terms on opposite sides. We also notice that

$$L_\alpha(m_1)b_2 = L_\alpha(m_2)b_1 - (x-\alpha)b_3$$

So if $L_\alpha(m_1) \neq 0$ then $\{b_1, b_3\}$ forms a basis for $\widetilde{M_{L_\alpha}}$, while if $L_\alpha(m_1) = 0$ then $\{b_2, b_3\}$ forms a basis. In both cases by construction the sets constructed have leading terms on opposite sides and thus by proposition 1 form a Gröbner basis. □

We want to use these results to solve the following problems.

- solve the key equation with one additional erasure
- solve the key equation with one fewer erasure
- solve the key equation with one additional syndrome
- interpolate a rational function through one additional point

For all of these problems we assume we have a Gröbner basis for the module of solutions and want to update the solution to incorporate an additional constraint. We will see how each of the problems can be characterized as the submodule given by the kernel of a suitable linear constraint.

We remark that the set $\{(s,1),(x^t,0)\}$ is a basis for the module M of solutions of the key equation

$$p \equiv qs (\mathrm{mod} x^t), \quad \text{where } p, q, s \in F[x]$$

Moreover if we have erasures and ψ, is the polynomial reperesenting them, then we can consider as a basis the set $\{(s\psi,\psi),(x^t,0)\}$. If this is the case then $discr(M) = x^t\psi$.

We will assume that the erasures are such that ψ is a square-free polynomial and $\psi(0) \neq 0$. We will say that an erasure at α is simple if α is a non-zero simple root of the erasure polynomial. To add or remove an erasure at $\alpha \in F^*$ is equivalent to multiply or divide ψ by $(x-\alpha)$. Our assumption that ψ is square-free implies that whenever we remove an erasure at α, then $(x-\alpha)||discr(M)$ (i.e. α is also a simple root of $discr(M)$).

3. Adding one erasure

Let $b = q\psi \in F[x]$ be the product of the error locator and erasure polynomial. Given $\alpha \in F^*$ such that $\psi(\alpha) \neq 0$ and given a Gröbner basis for the module

$$M = \{(a,b) \in A^2 \,|\, a \equiv bs \bmod x^t\}$$

we wish to find a basis for:

$$\widetilde{M} = \{(a,b) \in M \,|\, b(\alpha) = 0\}$$

i.e. we want to identify the submodule of M corresponding to error locator polynomials which vanish at α, so we have an additional erasure at α.

We can identify $\widetilde{M}$ as the kernel of a linear constraint defining $L_\alpha(a,b) = b(\alpha)$. By definition then $\mathrm{Ker}(L_\alpha) = \widetilde{M}$ and Proposition 1.5 furnishes a basis $\{n_1, n_2\}$ for $\widetilde{M}$.

Using the definitions of $\{n_1, n_2\}$ it easy to see that $discr(\widetilde{M}) = (x-\alpha)discr(M)$.

4. Removing one erasure

In this section we want to show how to "remove an erasure", i.e. given a module $\widetilde{M}$, with a simple erasure at α, we want to construct a module M such that $\widetilde{M} \subset M$ and $\widetilde{M}$ is obtained from M by imposing precisely an additional erasure at α.

Since $\widetilde{M}$ has an erasure at α we have that:

$$\widetilde{M} \subset \{(a,b) \in A^2 \,|\, a \equiv bs \bmod x^t \text{ and } b(\alpha) = 0\}$$

and our assumption that the erasure is a simple implies that $(x-\alpha)||discr(\widetilde{M})$.

Consider the map $\phi_\alpha : A^2 \mapsto A^2$ given by

$$\phi_\alpha(a,b) = ((x-\alpha)a, (x-\alpha)b),$$

and define

$$M = \{(a,b) \in A^2 \,|\, \phi_\alpha(a,b) \in \widetilde{M}\},$$

$$N = \{(a,b) \in M \,|\, b(\alpha) = 0\},$$

$$\widetilde{M_0} = \{(a,b) \in \widetilde{M} \,|\, a(\alpha) = 0\}.$$

By definition $\widetilde{M_0} \subset \widetilde{M} \subset N$, $\widetilde{M_0} = \phi_\alpha(M) = (x-\alpha)M$ and N is the module obtained from M by adding an erasure at α.

We want to show that $\widetilde{M} = N$. Since $\widetilde{M} \subset N$ we can use Proposition 1.3 and property (ii) of the discriminant. We have

- $discr(N) = (x-\alpha)discr(M)$ and $discr(\widetilde{M_0}) = (x-\alpha)discr(\widetilde{M})$, since N (resp. $\widetilde{M_0}$) is obtained from M (resp. $\widetilde{M}$) by adding a proper linear constraint at α.
- $discr((x-\alpha)M) = (x-\alpha)^2 discr(M)$.

From these releation then it follows that:

$$(x-\alpha)^2 discr(M) = discr(\widetilde{M_0}) = (x-\alpha) discr(\widetilde{M})$$

and hence that

$$discr(N) = (x-\alpha) discr(M) = discr(\widetilde{M})$$

which guarantees that $\widetilde{M} = N$.

Finally we remark that, by definition, M is the biggest submodule of A^2 such that by adding an erasure produces the given module $\widetilde{M}$.

We remark also that in order to construct a Gröbner basis for M we can use Proposition 1.5. Since $\widetilde{M_0} = (x-\alpha)M$, at first we construct a basis $\{n_1, n_2\}$ for $\widetilde{M_0}$ from a basis of $\widetilde{M}$ using proposition 1.5 with the linear constraint $L_\alpha(a,b) = a(\alpha)$ for $(a,b) \in \widetilde{M}$. Thus $\{\frac{n_1}{(x-\alpha)}, \frac{n_2}{(x-\alpha)}\}$ provides a basis for M.

5. Adding one syndrome

In this case we want to solve the congruence for a higher power of x, i.e. we have a basis for the module

$$M = \{(a,b) \in A^2 \,|\, a \equiv bs \bmod x^n\}$$

and we want to solve the same congruence with n replaced by $n+1$. $(a,b) \in M$ implies that $x^n | a - bs$. Consider the A-linear map $L : (a,b) \mapsto \frac{a-bs}{x^n}$ and then the F-linear map L_0 which is L followed by evaluation at 0. Thus we have that $\mathrm{Ker}(L_0)$ gives the submodule of M which satisfies $a \equiv bx \bmod x^{n+1}$. Although (1) also solves this problem using Gröbner theory, we obtain an automatic proof of correctness via linear constraints.

6. Interpolating through one additional point

The Berlekamp-Welch decoding algorithm leads to the following rational function interpolation problem, we are given $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ with $x_i, y_i \in F$ and we have a basis for the module

$$M = \{(a,b) \in A^2 \,|\, a(x_i) = y_i b(x_i) \text{ for } 1 \le i \le n\}$$

Given $x_{n+1}, y_{n+1} \in F$ we want to identify the submodule $\{(a,b) \in M \,|\, a(x_{n+1}) = y_{n+1} b(x_{n+1})\}$. Here we define the A-linear map $L : (a,b) \mapsto a - y_{n+1} b$ and then the F-linear map $L_{x_{n+1}}$ which is L followed by evaluation at x_{n+1}. Thus we have that $\mathrm{Ker}(L_{x_{n+1}})$ gives the submodule of M which satisfies $a(x_{n+1}) = y_{n+1} b(x_{n+1})$. Although (3) also gives a Gröbner approach for solving this problem, our solution is more direct and simpler using our notion of linear constraints.

7. Burst correction

An error burst is defined as a contiguous sequence of symbols potentially in error. Assuming the existence of such a burst at a particular starting location is the same as imposing erasures and thus allows one to correct additional errors as long as more than half of the locations in the burst are in fact in error. If we have at most r errors which lie outside a fixed contiguous burst sequence of length b, then we can correct all the errors as long as $b + 2r \leq t$ where t is the number of syndromes. If we require $b+2r < t$ then we can use the extra constraint to search for candidate burst start locations. This search can be conducted efficiently by sliding an erasure burst of length b across the received block of symbols. Assuming we have computed the key equation solution module corresponding to an erasure burst starting at the first position, we can successively slide this burst one position to the right by removing the erasure at the beginning of the burst and adding a new erasure at the end. For each candidate burst start location, we check to see if the the error locator from our module generators has degree at most $r + b$ and divides $x^{(l-1)} - 1$ where l is the order of our symbol field. This test guarantees that the roots of the error locator polynomial are valid positions within our symbol block. Each position where the error locator passes this test gives us a candidate codeword. Thus we can produce a list of feasible codewords to be passed to some additional discriminating procedure. Since burst errors are in fact a common cause of decoding failure in many practical situations, this can provide a successful recovery from decoder failure. Note that our unified framework for adding and removing erasures has allowed us to develop an incremental decoding algorithm which is much more efficient than the classical approach of separately constructing a burst of erasures at each possible starting location.

8. Conclusions

We have shown that using the framework of Gröbner bases for modules along with the notion of submodules satisfying linear constraints, allows a unified derivation of many incremental decoding algorithms. This unified framework allows us to guarantee that the various individual algorithms can be intermixed, yielding a novel approach for burst error correction.

Since the use decoding modules allows us to parameterize the space of solutions to the key equation, one can attempt to explore this solution space to generate a list of candidate codewords whose distance from the received data exceeds the error correcting radius of the code. This is a possible direction for future research.

References

[1] P.Fitzpatrick, "On the Key Equation", *IEEE Trans. on Information Theory*, vol 41, no 5, 1995, pp. 1290–1302.

[2] P.Fitzpatrick, "Errors-and-erasures decoding of BCH codes", *IEE Proc. Commun.*, Vol 146, No 2, 1999, pp. 79–81.

[3] S.M.Jennings, "Gröbner basis view of Welch–Berlekamp algorithm for Reed–Solomon codes", *IEE Proc. Commun.*, Vol 142, No 6, Dec. 1995, pp. 349–351.

[4] N. Kamiya, "On multisequence shift register synthesis and generalized-minimum-distance decoding of Reed-Solomon codes", *Finite Fields and Their Applications*, vol 1, no. 4, pp. 440-457, Oct. 1995

[5] R. Kötter, "Fast Generalized Minimum–Distance Decoding of Algebraic–Geometry and Reed–Solomon codes", *IEEE Trans. on Information Theory*, vol 42, no. 3, May 1996, pp. 721–737.

[6] S.Lang, *Algebra*, Addison-Wesley, 1984.

*

On Inverse Systems and Squarefree Decomposition of Zero-Dimensional Polynomial Ideals

To Bruno Buchberger on the occasion of his sixtieth birthday

WERNER HEIß[1], ULRICH OBERST[2] AND FRANZ PAUER[3]

Institut für Mathematik, Universität Innsbruck, Technikerstr. 25, A-6020 Innsbruck, Austria.

[1]*Werner.Heiss@uibk.ac.at*

[2]*Ulrich.Oberst@uibk.ac.at*

[3]*Franz.Pauer@uibk.ac.at*

Abstract

We show how to compute a basis of the inverse system of an arbitrary zero-dimensional ideal over an arbitrary field and present an algorithm to compute the squarefree decomposition of these ideals.

KEYWORDS: inverse system, polynomial ideal, squarefree decomposition

1. Introduction

Let F be an arbitrary field, n a positive integer, $F[s] := F[s_1, \ldots, s_n]$ the polynomial algebra in n indeterminates. Let I be an ideal in $F[s]$ with radical $R := \sqrt{I}$. We assume that I is zero-dimensional or, equivalently, that the algebra $F[s]/I$ is finite-dimensional as F-vector space.

By $I^\perp$ we denote the F-vector space of all linear functions $\varphi \in Hom_F(F[s], F)$ whose kernel contains I, i.e. $\varphi|_I = 0$. Then $I^\perp$ is the *inverse system* (7) of I. A *dual basis* of I is an F-basis of the vector space $I^\perp$.

In (9; 10; 11; 12; 13; 16) algorithms to compute a dual basis of I have been presented under the assumption that the set of zeroes of I is known and contained in F^n.

*This work was supported by the Austrian FWF through grant P15031.

In section 2 we show how to compute a dual basis of an arbitrary zero-dimensional ideal over an arbitrary fi eld. This method has its origin in (14) (see also (15)).

The inverse system $I^\perp$ is an $F[s]$-submodule of $Hom_F(F[s], F)$. Section 3 contains an algorithm to compute a system of $F[s]$-generators of $I^\perp$ which has minimal length if I is primary with rational radical.

In section 4 we generalise the well-known notion of squarefree decomposition of a univariate polynomial to the case of zero-dimensional ideals.

In section 5 we apply results of the previous sections to develop an algorithm to compute this decomposition. As in the case of univariate polynomials this decomposition can be computed without any assumptions on the fi eld and without knowing the primary decomposition or the zeroes of I.

The algorithms of this paper use Buchberger's theory of Gröbner bases (see (2; 3; 4)). We used Maple 8 and CoCoA 4 to compute the examples.

The inverse system $I^\perp$ is a special case of a multi-dimensional system or behaviour which are studied in (14; 15) and many other papers in system theory.

The results of sections 2 and 3 have been presented at the Rhine Workshop on Computer Algebra in Mannheim (6).

Finally, we announce results which will be contained in an extended version of this paper. We give a formula for the minimal length of a system of $F[s]$-generators of $I^\perp$ and a characterisation of this number in terms of the socle of the $F[s]$-module $F[s]/I$. If the primary decomposition of I is known a system of $F[s]$-generators of minimal length can be computed. Moreover, we describe the $F[s]$-module $I^\perp$ by generators and relations and give an algorithm to compute the coeffi cients of an element of $I^\perp$ with respect to a given system of generators. We also construct all commutative Frobenius algebras with their Frobenius homomorphism or residue.

2. An F-basis of $I^\perp$

Let $\leq$ be a term order on $\mathbb{N}^n$ and let $deg(g) \in \mathbb{N}^n$ be the degree of $g \in F[s]$ with respect to $\leq$. We denote by

$$\Gamma := \mathbb{N}^n \setminus deg(I)$$

the complement in $\mathbb{N}^n$ of the set of all degrees of non-zero polynomials in I. Since I is zero-dimensional the set Γ is fi nite.

Then

$$F[s] = I \oplus \bigoplus_{\gamma \in \Gamma} F s^\gamma.$$

Hence we get elements of $I^\perp$ by extending linear maps from $\bigoplus_{\gamma\in\Gamma} Fs^\gamma$ to F trivially to $I \oplus \bigoplus_{\gamma\in\Gamma} Fs^\gamma$. Thus the map

$$\begin{aligned} Hom_F(\bigoplus_{\gamma\in\Gamma} Fs^\gamma, F) &\longrightarrow I^\perp \subseteq Hom_F(I \oplus \bigoplus_{\gamma\in\Gamma} Fs^\gamma, F) \\ h &\longmapsto 0 \oplus h \end{aligned}$$

is F-linear and bijective. In particular, the F-vector space $I^\perp$ is finite-dimensional and its dimension is $card(\Gamma)$, the number of elements of Γ.
Since the familiy $(s^\alpha)_{\alpha\in\mathbb{N}^n}$ is an F-basis of $F[s]$, any linear function
$\varphi : F[s] \to F$ is uniquely determined by the family $(\varphi(s^\alpha))_{\alpha\in\mathbb{N}^n}$ in F. If φ is an element of $I^\perp$, it is sufficient to know the finite familiy$(\varphi(s^\gamma))_{\gamma\in\Gamma}$. The following theorem tells us how to compute the value $\varphi(s^\alpha)$ of $\varphi \in I^\perp$ at s^α for arbitrary $\alpha \in \mathbb{N}^n$.

Using a Gröbner basis of I, one can compute the normal form

$$\mathrm{nf}(g) \in \bigoplus_{\gamma\in\Gamma} Fs^\gamma$$

of $g \in F[s]$ such that $g - \mathrm{nf}(g) \in I$.

Theorem 1. *((14, 5.42/43/44), (15, Th. 5))*
Let $\alpha \in \mathbb{N}^n,\ \varphi \in I^\perp$, *and* $\mathrm{nf}(s^\alpha) = \sum_{\gamma\in\Gamma} c_\gamma s^\gamma$.
Then

$$\varphi(s^\alpha) = \sum_{\gamma\in\Gamma} c_\gamma \varphi(s^\gamma) \in F.$$

Thus we obtain the value $\varphi(s^\alpha)$ *for arbitrary* $\alpha \in \mathbb{N}^n$ *by computing the normal form* $\mathrm{nf}(s^\alpha)$ *of* s^α.

Proof: Since $s^\alpha - \mathrm{nf}(s^\alpha) \in I$ we have $\varphi(s^\alpha - \mathrm{nf}(s^\alpha)) = 0$, hence

$$\varphi(s^\alpha) = \varphi(\mathrm{nf}(s^\alpha) + (s^\alpha - \mathrm{nf}(s^\alpha))) = \varphi(\mathrm{nf}(s^\alpha)) = \sum_{\gamma\in\Gamma} c_\gamma \varphi(s^\gamma). \ \square$$

Let $\gamma \in \Gamma$. By e_γ we denote the uniquely determined F-linear map $e_\gamma \in I^\perp$ with $e_\gamma(s^\gamma) = 1$ and $e_\gamma(s^\alpha) = 0$, for all $\alpha \in \Gamma \setminus \{\gamma\}$.
Since I is a zero-dimensional ideal, the family $(e_\gamma)_{\gamma\in\Gamma}$ is an F-basis of $I^\perp$. By Theorem 1 we have

$$\mathrm{nf}(s^\alpha) = \sum_{\gamma\in\Gamma} e_\gamma(s^\alpha) s^\gamma.$$

In the sequel we always represent functions $\varphi \in I^\perp$ by the finite family$(\varphi(s^\gamma))_{\gamma\in\Gamma}$, i.e. by the family of coordinates of φ with respect to the basis $(e_\gamma)_{\gamma\in\Gamma}$.

Example 2. Let $I :=_{\mathbb{Q}[s_1,s_2]}< s_2^4, -s_2^3 + s_1s_2^2, s_2s_1^2, s_1^3 - s_2^2 + s_2s_1 >$ and let $\leq$ be the graded lexicographical term order with $s_1 > s_2$. Then

$$\Gamma = \{(0,0), (1,0), (0,1), (2,0), (1,1), (0,2), (0,3)\},$$

and the family $(e_\gamma)_{\gamma\in\Gamma}$ is a basis of $I^\perp$. We compute the values of $e_{(0,3)}$ for all $\alpha \in \mathbb{N}^n$. By definition$e_{(0,3)}(s_2^3) = 1$ and $e_{(0,3)}(s^\alpha) = 0$, for all $\alpha \in \Gamma \setminus \{(0,3)\}$. Hence

$$e_{(0,3)}(s_1s_2^2) = e_{(0,3)}(\mathrm{nf}(s_1s_2^2)) = e_{(0,3)}(s_2^3) = 1$$
$$e_{(0,3)}(s_1^4) = e_{(0,3)}(\mathrm{nf}(s_1^4)) = e_{(0,3)}(s_2^3) = 1 \quad .$$

Since $s^\alpha \in I$ for all $\alpha \in \mathbb{N}^n \setminus (\Gamma \cup \{(1,2), (4,0)\})$ we get $\mathrm{nf}(s^\alpha) = 0$, hence

$$e_{(0,3)}(s^\alpha) = 0 \text{ , for } \alpha \in \mathbb{N}^n \setminus (\Gamma \cup \{(1,2), (4,0)\}).$$

The assertions in the following lemma are well-known and immediate consequences of the defi nition of $I^\perp$.

Lemma 3. *Let I_1, I_2 be ideals in $F[s]$. Then*

$$(I_1 + I_2)^\perp = {I_1}^\perp \cap {I_2}^\perp \text{ and}$$
$$(I_1 \cap I_2)^\perp = I_1^\perp + I_2^\perp.$$

If I_1 and I_2 are comaximal (i.e. $I_1 + I_2 = F[s]$) then

$$(I_1 \cap I_2)^\perp = I_1^\perp \oplus I_2^\perp.$$

Definition 4. Let W be an F-subspace of $Hom_F(F[s], F)$. By $W^\perp$ we denote the F-vector space of all polynomials $g \in F[s]$ such that $\varphi(g) = 0$, for all $\varphi \in W$.

Lemma 5. *Let $W \subseteq I^\perp$ be an F-subspace of $I^\perp$. Then*

$$W^\perp = I \oplus (W^\perp \cap \bigoplus_{\gamma\in\Gamma} Fs^\gamma)\,,$$

hence $W^\perp$ is uniquely determined by the F-subspace $W^\perp \cap \bigoplus_{\gamma\in\Gamma} Fs^\gamma$. In particular we obtain $(I^\perp)^\perp = I$ and $(W^\perp)^\perp = W$.

Proof: Since $I \subseteq W^\perp$ and $W \subseteq I^\perp$ the assertion follows immediately from

$$F[s] = I \oplus \bigoplus_{\gamma\in\Gamma} Fs^\gamma$$

and the modular law. □

Remark 6. If $\varphi_1, ..., \varphi_\ell$ is an F-basis of W, then an F-basis of $W^\perp \cap \bigoplus_{\gamma\in\Gamma} Fs^\gamma$ can be computed by solving the system

$$\sum_{\gamma\in\Gamma} c_\gamma \varphi_i(s^\gamma) = 0,\ 1 \le i \le \ell\,,$$

of linear equations for $c_\gamma,\ \gamma \in \Gamma$.

Example 7. Let I be the ideal

$$I =_{\mathbb{Q}[s_1,s_2]} < 5s_1s_2 - 5s_1 - 3s_2^2 + 3s_2, s_2^3 - 6s_2^2 + 5s_2,$$
$$5s_1^6 - 15s_1^5 + 15s_1^4 - 5s_1^3 - 54s_2^2 + 54s_2 > .$$

With respect to the graded lexicographical term order with $s_1 > s_2$ we get

$$\Gamma = \{(0,0), (0,1), (1,0), (0,2), (2,0), (3,0), (4,0), (5,0)\}$$

and the basis

$$(e_{(0,0)}, e_{(0,1)}, e_{(1,0)}, e_{(0,2)}, e_{(2,0)}, e_{(3,0)}, e_{(4,0)}, e_{(5,0)})$$

of $I^\perp$. Let $W \subseteq I^\perp$ be subspace generated by

$$\begin{aligned}\varphi_1 &= e_{(0,0)} + e_{(0,1)} + e_{(0,2)} \text{ and}\\ \varphi_2 &= e_{(1,0)} + e_{(2,0)} + e_{(3,0)} + e_{(4,0)} + e_{(5,0)} \ .\end{aligned}$$

Then $W^\perp \cap \bigoplus_{\gamma\in\Gamma} \mathbb{Q}s^\gamma$ is the set of all polynomials $g = \sum_{\gamma\in\Gamma} c_\gamma s^\gamma$ such that

$$\begin{aligned}\varphi_1(g) &= c_{(0,0)} + c_{(0,1)} + c_{(0,2)} = 0\\ \varphi_2(g) &= c_{(1,0)} + c_{(2,0)} + c_{(3,0)} + c_{(4,0)} + c_{(5,0)} = 0 \quad .\end{aligned}$$

Solving this linear equation we get

$$W^\perp \cap \bigoplus_{\gamma\in\Gamma} \mathbb{Q}s^\gamma =_{\mathbb{Q}} < s_2 - 1, s_2^2 - 1, s_1^2 - s_1, s_1^3 - s_1, s_1^4 - s_1, s_1^5 - s_1 >$$

and hence

$$W^\perp = I \oplus_{\mathbb{Q}} < s_2 - 1, s_2^2 - 1, s_1^2 - s_1, s_1^3 - s_1, s_1^4 - s_1, s_1^5 - s1 > .$$

3. $I^\perp$ as $F[s]$-Module

There is a natural $F[s]$-module-structure on $Hom_F(F[s], F)$: for $f, g \in F[s]$ and $\varphi \in Hom_F(F[s], F)$ we defi ne

$$(f \circ \varphi)(g) := \varphi(fg).$$

Since I is an ideal, $I^\perp$ is an $F[s]$-submodule of $Hom_F(F[s], F)$. On the other hand, if W is an $F[s]$-submodule of $Hom_F(F[s], F)$, then $W^\perp$ is an ideal in $F[s]$. Note that I annihilates $I^\perp$, i.e. $I \circ I^\perp = 0$. Since $(I^\perp)^\perp = I$ we have $I = (0 : I^\perp)$, where $(0 : I^\perp)$ is the annihilator of $I^\perp$, i.e. the ideal $\{g \in F[s] \mid g \circ I^\perp = 0\}$.

We quote the Lemma of Krull-Nakayama (see (8, Th. 2.2, Th 2.3)), which in our special case has the form

Lemma 8. *Let X be a finitely generated $F[s]$-module annihilated by I, i.e. $I \circ X = 0$. Let Y be a submodule and $x_1, ..., x_m$ elements of X.*

(1) If $X = Y + R \circ X$ then $Y = X$.

(2) The elements $x_i, 1 \le i \le m$, generate X if and only if their residue classes $\overline{x_i} := x_i + R \circ X, 1 \le i \le m$, generate $X/(R \circ X)$.

(3) The elements $x_i, 1 \le i \le m$, are a minimal system of generators of X if and only if the elements $\overline{x_i}, 1 \le i \le m$, are this for $X/(R \circ X)$.

(4) Assume in addition that the ideal I is primary or, in other terms, that $F[s]/R$ is a field. Then any minimal system of generators of X is a system of generators of minimal length. The elements $x_i, 1 \leq i \leq m$, are a minimal system of generators of X if and only if the elements $\overline{x_i}, 1 \leq i \leq m$, are an $F[s]/R$-basis of $X/(R \circ X)$.

Corollary 9. *Let V be an F-subspace of $I^\perp$ such that $I^\perp = V \oplus R \circ I^\perp$. Then:*

(1) Each basis of V is a system of generators of $I^\perp$, but in general not a minimal one. In other terms $I^\perp = F[s] \circ V$. Moreover, $R \circ I^\perp = R \circ V$ and $I^\perp = V \oplus R \circ V$.

(2) If I is primary and R is rational, i.e. $F[s]/R \cong F$, then any F-basis of V is a system of $F[s]$-generators of $I^\perp$ of minimal length.

Proof:

(1) The decomposition $X = V \oplus R \circ X$ induces the F-isomorphism $V \cong X/(R \circ X)$. Hence any basis of V is particularly a system of generators of $X/(R \circ X)$ and therefore of X by Lemma 8.

(2) Follows directly from Lemma 8. □

Lemma 10. *Let $J \subseteq F[s]$ be an ideal, G a system of generators of J and B an F-basis of $I^\perp$. Then $\{g \circ b \mid g \in G, b \in B\}$ is a system of generators of the F-vector space $J \circ I^\perp$.*

Proof:

$$F \circ (G \circ B) = G \circ F \circ B = G \circ F[s] \circ B = \\ = F[s] \circ G \circ FB = J \circ I^\perp \quad . \quad \square$$

Theorem 11. *The following algorithm computes a subset E of $\{e_\gamma \mid \gamma \in \Gamma\}$ such that*

$$I^\perp =_F < E > \bigoplus R \circ_F < E > .$$

- *Compute a system G of generators of the radical R of I.*
- *For $f = \sum_\alpha a_{f,\alpha} s^\alpha \in G$ and $\gamma \in \Gamma$ compute the normal form of $s^{\alpha+\beta}$ with respect to I and thus $e_\gamma(s^{\alpha+\beta})$ for all $\alpha \in supp(f)$ and $\beta \in \Gamma$. Then*

$$f \circ e_\gamma = \sum_{\beta \in \Gamma} \sum_{\alpha \in supp(f)} (a_{f,\alpha} e_\gamma(s^{\alpha+\beta})) e_\beta.$$

- *Choose an F-basis of $_F < f \circ e_\gamma \mid \gamma \in \Gamma, f \in G >$ and complete it by a subset E of $\{e_\gamma \mid \gamma \in \Gamma\}$ to a basis of $I^\perp$.*

Proof: By Lemma 10 we have

$$R \circ I^{\perp} =_F < f \circ e_\gamma \mid \gamma \in \Gamma, f \in G > .$$

Let $d_{f,\gamma,\beta} \in F$ such that $f \circ e_\gamma = \sum_{\beta \in \Gamma} d_{f,\gamma,\beta} e_\beta$. Then

$$d_{f,\gamma,\beta} = (f \circ e_\gamma)(s^\beta) = \sum_{\alpha \in supp(f)} a_{f,\alpha}(s^\alpha \circ e_\gamma)(s^\beta) =$$
$$= \sum_{\alpha \in supp(f)} a_{f,\alpha} e_\gamma(s^{\alpha+\beta}).$$

Now the assertion follows from Corollary 9, (1). □

Remark 12. In order to compute the radical R of I, one usually first determines univariate polynomials $g_i \in F[s_i]$, $1 \leq i \leq n$, such that R is generated by I and $g_1, ..., g_n$. In the preceding algorithm we can replace G by $\{g_1, ..., g_n\}$ since $f \circ e_\gamma = 0$ for $f \in I$.

Example 13. Let $I :=_{\mathbb{Q}[s]} < s_2^5 + 4s_2^4 + 4s_2^3, 2s_1s_2^3 + s_2^4 + 4s_1s_2^2 + 2s_2^3,$
$4s_1^2s_2 + 4s_1s_2^2 + s_2^3, 4s_1^3 - s_2^3 - 3s_1s_2^2 >$. The radical of I is generated by the set $G := \{s_2^2 + 2s_2, 2s_1 + s_2\}$ and the set $\{s^\gamma \mid \gamma \in \Gamma\}$ (using the graded lexicographical term order with $s_1 > s_2$) is $\{1, s_1, s_2, s_1^2, s_1s_2, s_2^2, s_1s_2^2, s_2^3, s_2^4\}$. We compute

$$\{f \circ e_\gamma \mid f \in G, \gamma \in \Gamma\} =$$
$$\{0, 2e_{(0,0)}, 2e_{(1,0)}, e_{(0,0)} + 2e_{(0,1)}, e_{(1,0)},$$
$$e_{(0,1)} + 1/2e_{(2,0)} - e_{(1,1)} + 2e_{(0,2)} + 2e_{(1,2)} - 4e_{(0,3)} + 8e_{(0,4)},$$
$$1/4e_{(2,0)} - 1/2e_{(1,1)} + e_{(0,2)} + e_{(1,2)} - 2e_{(0,3)} + 4e_{(0,4)},$$
$$2e_{(0,0)}, e_{(0,0)}, 2e_{(1,0)}, e_{(1,0)} + 2e_{(0,1)}, e_{(0,1)},$$
$$1/2e_{(2,0)} - e_{(1,1)} + 2e_{(0,2)} + 2e_{(1,2)} - 4e_{(0,3)} + 8e_{(0,4)},$$
$$1/4e_{(2,0)} - 1/2e_{(1,1)} + e_{(0,2)} + e_{(1,2)} - 2e_{(0,3)} + 4e_{(0,4)}\}.$$

The set

$$\{e_{(0,0)}, e_{(1,0)}, e_{(0,1)}, e_{(2,0)} - 2e_{(1,1)} + 4e_{(0,2)} + 4e_{(1,2)} - 8e_{(0,3)} + 16e_{(0,4)}\}.$$

is a basis of $_{\mathbb{Q}} < f \circ e_\gamma \mid \gamma \in \Gamma, f \in G >= R \circ I^{\perp}$. Then

$$E = \{e_{(1,1)}, e_{(0,2)}, e_{(1,2)}, e_{(0,3)}, e_{(2,0)}\}.$$

Lemma 14. *(see (16, Lemma 2.2.1)) Let J be any ideal in $F[s]$ and let $(I : J)$ be the ideal quotient of I by J. Then*

$$(I : J)^{\perp} = J \circ I^{\perp} .$$

Proof: According to Lemma 5 we have $(J \circ I^{\perp})^{\perp\perp} = J \circ I^{\perp}$. Hence it suffi ces to prove that $(I : J) = (J \circ I^{\perp})^{\perp}$. But

$$\begin{aligned}(I : J) &= \{f \in F[s] \mid fg \in I, \text{ for all } g \in J\} = \\ &= \{f \in F[s] \mid \varphi(fg) = 0, \text{ for all } g \in J \text{ and all } \varphi \in I^{\perp}\} = \\ &= \{f \in F[s] \mid (g \circ \varphi)(f) = 0, \text{ for all } g \in J \text{ and all } \varphi \in I^{\perp}\} = \\ &= (J \circ I^{\perp})^{\perp} \quad . \quad \square\end{aligned}$$

Let

$$I = \bigcap_{i=1}^{\ell} Q_i$$

be the minimal primary decomposition of the zero-dimensional ideal I with primary ideals Q_i and associated radical ideals P_i, $1 \leq i \leq \ell$. Since I is zero-dimensional the ideals P_i are maximal and, indeed, exactly the maximal ideals containing I. Moreover the primary decomposition of R is

$$R = \bigcap_{i=1}^{\ell} P_i \, .$$

The ideals

$$Q_j \text{ and } \bigcap_{i,i\neq j} Q_i \text{ resp.}$$
$$P_j \text{ and } \bigcap_{i,i\neq j} P_i$$

are comaximal, $1 \leq j \leq \ell$, hence

$$\bigcap_{i=1}^{\ell} Q_i = \prod_{i=1}^{\ell} Q_i \text{ and}$$
$$\bigcap_{i=1}^{\ell} P_i = \prod_{i=1}^{\ell} P_i \, .$$

Theorem 15.

(1) The minimal primary decomposition of I induces

$$I^{\perp} = \bigoplus_{i=1}^{\ell} Q_i^{\perp} \, .$$

$Q_i^{\perp}$ is an $F[s]-$submodule of $I^{\perp}$, $1 \leq i \leq \ell$. If an F-basis $\varphi_1, ..., \varphi_m$ of $Q_i^{\perp}$ is known, Lemma 5 furnishes the primary component

$$Q_i = (Q_i^{\perp})^{\perp} = \{g \in F[s] \mid \varphi_j(g) = 0, \ 1 \leq j \leq m\}$$

by solving a system of linear equations (compare (13, Th. 8)).

(2) If $i \neq j$ then

$$P_i \circ Q_j^{\perp} = Q_j^{\perp},$$

moreover

$$R \circ I^{\perp} = \bigoplus_{j=1}^{\ell} P_j \circ Q_j^{\perp} \quad .$$

Proof:

(1) Follows from Lemma 3.

(2) Obviously $R \circ I^{\perp} = \bigoplus_{j=1}^{\ell} R \circ Q_j^{\perp}$. The module $Q_j^{\perp}$ is annihilated by Q_j. If $i \neq j$ then Q_j is comaximal to P_i. We conclude

$$P_i \circ Q_j^{\perp} = (P_i + Q_j) \circ Q_j^{\perp} = F[s] \circ Q_j^{\perp} = Q_j^{\perp}, \text{ hence}$$

$$R \circ Q_j^{\perp} = (\prod_{i=1}^{\ell} P_i) \circ Q_j^{\perp} = P_j \circ Q_j^{\perp} \text{ and}$$

$$R \circ I^{\perp} = \bigoplus_{j=1}^{\ell} R \circ Q_j^{\perp} = \bigoplus_{j=1}^{\ell} P_j \circ Q_j^{\perp} \quad . \ \square$$

4. Squarefree Decomposition of Zero-Dimensional Ideals

Recall that a polynomial $f \in F[s_1]$ is *squarefree* if it has no proper quadratic divisors, i.e. for any $g \in F[s_1]$ with g^2 dividing f we have $g \in F$.

Definition 16. ((5, 14.6.)) Let $f \in F[s_1]$ be a non-constant, monic polynomial. The squarefree decomposition of f is the (unique) sequence of monic squarefree pairwise coprime polynomials $(g_1, ..., g_m)$ such that $g_m \neq 1$ and

$$f = g_1 g_2^2 .... g_m^m \, .$$

Example 17. Let $f := s_1^4(s_1+1)^2(s_1-1)^2(s_1^2+1)^2(s_1^2+s_1+1) \in \mathbb{Q}[s_1]$. The squarefree decomposition of f is $(s_1^2+s_1+1, s_1^4-1, 1, s_1)$.

Note that if $f = \prod f_i^{e_i}$ is the irreducible factorization of f then we obtain the squarefree decomposition of f by collecting irreducible factors of the same exponent, i.e. the squarefree decomposition of f is

$$(\prod_{i,e_i=1} f_i, ..., \prod_{i,e_i=m} f_i) \, ,$$

where $m = \max(e_i)$. The important point is that these products can be computed without knowing the irreducible factors. This process is called the squarefree factorization of a polynomial. For further information we refer to (5, 14.6.).

Let again $I = \bigcap_{i=1}^{\ell} Q_i$ be the minimal primary decomposition of I and P_i be the radical of Q_i, $1 \leq i \leq \ell$.

Definition 18. For $1 \leq i \leq \ell$ let ε_i be the smallest natural number k such that $P_i^k \subseteq Q_i$. This number is called the *exponent of* Q_i. Let $m := \max\{\varepsilon_i \mid 1 \leq i \leq \ell\}$. Then m is the *exponent of* I, i.e. the least natural number k such that $R^k \subseteq I$.

Definition 19. For $1 \leq k \leq m$ we set

$$I_k := \bigcap_{i,\varepsilon_i=k} Q_i \quad \text{and}$$
$$R_k := \bigcap_{i,\varepsilon_i=k} P_i \; .$$

The *squarefree decomposition* of the zero-dimensional ideal I is the (unique) sequence

$$(R_1, ..., R_m) \; .$$

Note that if I has no primary component of exponent j then $R_j = F[s_1, ..., s_n]$.

Theorem 20.

(1) For $1 \leq i \leq \ell$ *we have* $Q_i = I + P_i^{\varepsilon_i}$.

(2) For $1 \leq k \leq m$ *we have* $I_k = I + R_k^k$.

(3) $I = \bigcap_{k=1}^m I_k = \bigcap_{k=1}^m (I + R_k^k)$.

Proof:

(1) See (1), ch. 8.

(2) For all j with $\varepsilon_j = k$ we have

$$I + (\bigcap_{i,\varepsilon_i=k} P_i)^k \subseteq I + P_j^k = Q_j \; ,$$

hence $I + (\bigcap_{i,\varepsilon_i=k} P_i)^k \subseteq \bigcap_{i,\varepsilon_i=k} Q_i$.
On the other hand, since

$$\bigcap_{i,\varepsilon_i=k} Q_i = \prod_{i,\varepsilon_i=k} Q_i \quad \text{and}$$
$$\bigcap_{i,\varepsilon_i=k} P_i = \prod_{i,\varepsilon_i=k} P_i$$

we have

$$\bigcap_{i,\varepsilon_i=k} Q_i = \prod_{i,\varepsilon_i=k} Q_i = \prod_{i,\varepsilon_i=k} (I + P_i^k) \subseteq I + \prod_{i,\varepsilon_i=k} P_i^k = I + (\bigcap_{i,\varepsilon_i=k} P_i)^k \; .$$

(3) Obvious. □

5. Computation of the Squarefree Decomposition

In this section we describe a method to compute the submodules $R_k^\perp$ of $I^\perp$. Then $R_k^{\perp\perp} = R_k$ (see Lemma 5) and we can compute I_k using Theorem 20, (2). The important point is that the ideals R_k and I_k can be computed without knowing the primary decomposition of I.

Consider the descending chain of $F[s]$-submodules of $I^\perp$

$$I^\perp \supseteq R \circ I^\perp \supseteq R^2 \circ I^\perp \supseteq ... \supseteq R^{m-1} \circ I^\perp \supseteq R^m \circ I^\perp.$$

Since m is the exponent of the radical R of I it is clear that $R^m \subseteq I$ and $R^m \circ I^\perp = 0$.

Lemma 21. *For $k \in \mathbb{N}$ and for $1 \leq i \leq \ell$ we have*

$$R^k \circ I^\perp = \bigoplus_{j=1}^{\ell} P_j^k \circ Q_j^\perp \cdot$$

Proof: The assertion is a direct consequence of Theorem 15. □

Lemma 22. *For $1 \leq i \leq \ell$ we have*

$$P_i^{\varepsilon_i - 1} \circ Q_i^\perp = P_i^\perp \, .$$

Proof: By definition of the exponent ε_i we have $P_i^{\varepsilon_i} \subseteq Q_i$ and $P_i^{\varepsilon_i - 1} \not\subseteq Q_i$. Hence $P_i \subseteq (Q_i : P_i^{\varepsilon_i - 1})$ and $(Q_i : P_i^{\varepsilon_i - 1}) \neq F[s]$. Since P_i is maximal, this implies $(Q_i : P_i^{\varepsilon_i - 1}) = P_i$. By Lemma 14 this implies the assertion. □

Lemma 23. *For $2 \leq k \leq m$*

$$R^{k-1} \circ I^\perp = R_k^\perp \bigoplus \bigoplus_{i, \varepsilon_i > k} P_i^{k-1} \circ Q_i^\perp.$$

Proof: By Lemma 21 we have

$$R^{k-1} \circ I^\perp = \bigoplus_{j=1}^{\ell} P_j^{k-1} \circ Q_j^\perp.$$

If $\varepsilon_i < k$ then by the definition of ε_i we have $P_i^{k-1} \subseteq Q_i$, hence $P_i^{k-1} \circ Q_i^\perp = 0$. Therefore

$$R^{k-1} \circ I^\perp = \bigoplus_{i, \varepsilon_i = k} P_i^{k-1} \circ Q_i^\perp \bigoplus \bigoplus_{i, \varepsilon_i > k} P_i^{k-1} \circ Q_i^\perp \quad .$$

By Lemma 22 we have

$$\bigoplus_{i, \varepsilon_i = k} P_i^{k-1} \circ Q_i^\perp = \bigoplus_{i, \varepsilon_i = k} P_i^\perp = R_k^\perp \quad . \quad \square$$

Theorem 24.

(1) $R_m^\perp = R^{m-1} \circ I^\perp$

(2) For $1 \leq k \leq m-1$ *we have*

$$R_k^\perp = (\prod_{j=k+1}^m R_j^{j-k+1}) \circ (R^{k-1} \circ I^\perp).$$

In particular, we can compute $R_k^\perp$ *recursively, starting with* $k = m$ *(and going down to k=1).*

Proof: (1) follows from Lemma 23 for $k = m$.

(2) By Lemma 23 we have

$$(\prod_{j=k+1}^m R_j^{j-k+1}) \circ (R^{k-1} \circ I^\perp) = (\prod_{j=k+1}^m R_j^{j-k+1}) \circ \bigoplus_{i,\varepsilon_i \geq k} P_i^{k-1} \circ Q_i^\perp =$$
$$= \bigoplus_{i,\varepsilon_i \geq k} ((\prod_{j=k+1}^m R_j^{j-k+1}) P_i^{k-1}) \circ Q_i^\perp.$$

For every i with $\varepsilon_i > k$ the ideal $((\prod_{j=k+1}^m R_j^{j-k+1}) P_i^{k-1})$ is contained in Q_i, hence we have

$$((\prod_{j=k+1}^m R_j^{j-k+1}) P_i^{k-1}) \circ Q_i^\perp = 0\,.$$

Thus

$$\bigoplus_{i,\varepsilon_i \geq k} ((\prod_{j=k+1}^m R_j^{j-k+1}) P_i^{k-1}) \circ Q_i^\perp = \bigoplus_{i,\varepsilon_i = k} ((\prod_{j=k+1}^m R_j^{j-k+1}) P_i^{k-1}) \circ Q_i^\perp\,.$$

For every i with $\varepsilon_i = k$ the ideals P_i^{k-1} and $(\prod_{j=k+1}^m R_j^{j-k+1})$ are coprime, hence $(\prod_{j=k+1}^m R_j^{j-k+1}) \not\subseteq P_i$ and by Theorem 15 we have

$$(\prod_{j=k+1}^m R_j^{j-k+1}) \circ Q_i^\perp = Q_i^\perp\,.$$

Therefore with Lemma 22 we get

$$((\prod_{j=k+1}^m R_j^{j-k+1}) P_i^{k-1}) \circ Q_i^\perp = P_i^{k-1}((\prod_{j=k+1}^m R_j^{j-k+1}) \circ Q_i^\perp) =$$
$$= P_i^{k-1} \circ Q_i^\perp = P_i^\perp\,.$$

Now

$$(\prod_{j=k+1}^m R_j^{j-k+1}) \circ (R^{k-1} \circ I^\perp) = \bigoplus_{i,\varepsilon_i = k} P_i^\perp = R_k^\perp.\quad \square$$

This theorem implies the following algorithm:

Algorithm 25. *Squarefree Decomposition of Zero-Dimensional Ideals*
Input: I a zero-dimensional ideal
R the radical of I
Output: $(R_1, ..., R_m)$ the squarefree decomposition of I

$S_0 := I^\perp$
$i := 0$
Repeat
$S_{i+1} := R \circ S_i$
$i := i + 1$
Until $S_i = 0$

$m := i$
$R_m^\perp := S_{m-1}$

For k from $m-1$ to 1
$R_{k+1} := (R_{k+1}^\perp)^\perp$
$R_k^\perp := \prod_{j=k+1}^{m} R_j^{j-k+1} \circ S_{k-1}$

Return $(R_1, ..., R_m)$

Example 26. Let I be the ideal in $\mathbb{Q}[s_1, s_2]$ generated by

$$5s_1s_2 - 5s_1 - 3s_2^2 + 3s_2, \;\; s_2^3 - 6s_2^2 + 5s_2 \text{ and}$$
$$5s_1^6 - 15s_1^5 + 15s_1^4 - 5s_1^3 - 54s_2^2 + 54s_2 \; .$$

We compute the set Γ (with respect to the total degree term order with $s_1 > s_2$) and obtain a basis of the dual space of I :

$$\left(e_{(0,0)}, e_{(0,1)}, e_{(1,0)}, e_{(0,2)}, e_{(2,0)}, e_{(3,0)}, e_{(4,0)}, e_{(5,0)}\right) .$$

The ideal R is generated by

$$5s_1s_2 - 5s_1 - 3s_2^2 + 3s_2, \quad -3s_2^2 + 3s_2 + 10s_1^2 - 10s_1 \text{ and}$$
$$s_2^3 - 6s_2^2 + 5s_2.$$

Now we compute $\mathbb{Q}-$bases for the $\mathbb{Q}[s_1, s_2]-$submodules $R^k \circ I^\perp$ for increasing k :

$$S_0 = I^\perp$$
$$S_1 = R \circ I^\perp =_{\mathbb{Q}} < e_{(0,0)} + e_{(0,1)} + e_{(0,2)}, e_{(1,0)}, e_{(2,0)} - e_{(4,0)} - 2e_{(5,0)},$$
$$\quad e_{(3,0)} + 2e_{(4,0)} + 3e_{(5,0)} >$$
$$S_2 = R^2 \circ I^\perp =_{\mathbb{Q}} < e_{(0,0)} + e_{(0,1)} + e_{(0,2)},$$
$$\quad e_{(1,0)} + e_{(2,0)} + e_{(3,0)} + e_{(4,0)} + e_{(5,0)} >$$
$$S_3 = R^3 \circ I^\perp = 0$$

hence the exponent m of I is 3 and

$$R_3^{\perp} = S_2 =_{\mathbb{Q}} < e_{(0,0)} + e_{(0,1)} + e_{(0,2)}, e_{(1,0)} + e_{(2,0)} + e_{(3,0)} + e_{(4,0)} + e_{(5,0)} >$$

is the dual space of the intersection of the associated maximal ideals of the primary components of exponent 3 of I. We compute

$$R_3 =_{\mathbb{Q}[s_1,s_2]} < s_1^2 - s_1, s_2 - 1 >$$

by solving a linear system of equations and obtain in the next step

$$R_2^{\perp} = R_3^2 \circ S_1 = \{0\}.$$

The ideal corresponding to $R_2^{\perp} = \{0\}$ is the entire polynomial ring

$$R_2 = \mathbb{Q}[s_1, s_2]\,,$$

hence there are no proper primary ideals of exponent 2 in the primary decomposition of I. We fi nally obtain the submodule $R_1^{\perp}$ by computing

$$\begin{aligned} R_1^{\perp} = R_2^2 R_3^3 \circ S_1 = R_3^3 \circ S_1 =_{\mathbb{Q}} < e_{(0,0)}, \\ e_{(0,1)} + \frac{3}{5}e_{(1,0)} + 5e_{(0,2)} + \frac{9}{5}e_{(2,0)} + \frac{27}{5}e_{(3,0)} + \frac{81}{5}e_{(4,0)} + \frac{243}{5}e_{(5,0)} > . \end{aligned}$$

The corresponding ideal is

$$R_1 =_{\mathbb{Q}[s_1,s_2]} < 5s_1 - 3s_2, s_2^2 - s_2 > .$$

Now we apply Theorem 20 and get

$$\begin{aligned} I_1 &= R_1 =_{\mathbb{Q}[s_1,s_2]} < 5s_1 - 3s_2, s_2^2 - s_2 > , \\ I_2 &= \mathbb{Q}[s_1, s_2]\,, \\ I_3 &=_{\mathbb{Q}[s_1,s_2]} < s_2 - 1, s_1^6 - 3s_1^5 + 3s_1^4 - s_1^3 > \ . \end{aligned}$$

References

[1] Becker, T., Weispfenning, V.: Gröbner bases. Springer-Verlag, New York, 1993.

[2] Buchberger, B.: Ein Algorithmus zum Auffi nden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal. Dissertation, Innsbruck (1965).

[3] Buchberger, B.: Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems. Aequationes Math. 4 (1970), 374-383.

[4] Buchberger, B, Winkler, F. (eds.): Gröbner bases and Applications. Cambridge University Press, Cambridge, 1998.

[5] von zur Gathen, J., Gerhard, J.: Modern Computer Algebra. Cambridge University Press, Cambridge, 1999.

[6] Heiß, W., Oberst, U., Pauer, F.: On dual spaces of polynomial ideals. Proceedings of the 8^{th} Rhine Workshop on Computer Algebra (2002), 195-207.

[7] Macaulay, F. S.: The algebraic theory of modular systems. Cambridge University Press, Cambridge, 1916.

[8] Matsumura, H.: Commutative ring theory. Cambridge University Press, Cambridge, 1986.

[9] Marinari, M. G., Möller, H. M., Mora, T.: Gröbner bases of ideals defined by functionals with an application to ideals of projective points. AAECC 4 (1993), 103-145.

[10] Marinari, M. G., Möller, H. M., Mora, T.: On multiplicities in polynomial system solving. Trans. Amer. Math. Soc. 348 (1996), 3283-3321.

[11] Möller, H. M., Mora, T.: Gröbner bases of ideals given by dual bases. Proc. ISSAC 91, ACM (1991), 55-63.

[12] Möller, H. M., Tenberg, R.: Multivariate polynomial system solving using intersections of eigenspaces. J. Symbolic Computation 30 (1999), 1-19.

[13] Mourrain, B.: Isolated points, duality and residues. Journal of pure and applied algebra 117&118 (1997), 469-493.

[14] Oberst, U.: Multidimensional Constant Linear Systems. Acta Appl. Math. 20 (1990), 1-175.

[15] Oberst, U., Pauer, F.: The Constructive Solution of Linear Systems of Partial Difference and Differential Equations with Constant Coefficients. Multidimensional Systems and Signal Processing 12 (2001), 253-308.

[16] Tenberg, R.: Duale Basen nulldimensionaler Ideale und Anwendungen. Shaker-Verlag, Aachen, 2000.

Two Paradigms of Learning

Wolfram Menzel[1] and Frank Stephan[2]

[1] *Institut für Logik, Komplexität und Deduktionssysteme, Universität Karlsruhe, 76128 Karlsruhe, Germany*

[2] *Mathematisches Institut, Universität Heidelberg, 69120 Heidelberg, Germany.* *Supported by: Deutsche Forschungsgemeinschaft (DFG) under Heisenberg grant Ste 967/1-1.*

Abstract

Two approaches to model learning phenomena are related to each other, the statistical and the inductive one. A central point where they differ is the used notion of convergence. We investigate whether and to which extent, in the world of computable functions over the natural numbers, the two respective types of convergence can be combined.

KEYWORDS: Statistical learning theory, inductive inference, "static" versus "algorithmic" mathematics

Prelude

Must it be Gröbner bases or Gödel numberings or a new didactics of mathematics, if one searches for a good topic referring to Bruno Buchberger's work? Certainly not, quite the contrary: It would be difficult to find anything interesting from mathematical computer science or from algorithmic mathematics that does *not* have such a reference. So when we just try to look – cautiously, humbly – at these two super-powers, mathematics and computer science, two patrons or demons or godheads who invisibly guide this meeting, look at their strange and exciting kind of relationship, somehow iridescent between dependency and autonomy, curiosity and caution, – if we take any essential question that comes from the joys or battles between those two we will with probability 1 find Bruno Buchberger engaged.

I will talk about a question which, I think, mirrors at a specific point that difficult mentioned relationship. It comes from learning theory. Learning is the finding of a finite object – a "pattern" or "rule" or "program" or "parameter value" – such that, by this find, a certain problem becomes better tractable or tractable at all.

The concern here is in fact an age-long and famous philosophical one, the *induction question*: How can one "infer", in a well-justified way, from finitely many data a general law which underlies them and thus governs an infinity of them? I will not, of course, deal with the immense and complicated philosophical discussions that arose and arise here, but will take a more pragmatic view, which is that of mathematical learning theory: Learning (induction) certainly *is* possible in appropriate situations (simply because it permanently happens), so that the task which matters is rather to determine

- what the right context to describe the meant phenomena is;
- precisely what, then, (successful) learning means;
- what necessary and/or sufficient conditions for the desired success are.

Various types, grades etc. of learning may of course result. Furthermore, we will here aim at *algorithms* being able to learn in the intended sense.

It is the question of the right context and definition of "learning" where, I think, a contrast between the classically mathematical and the algorithmic way of thinking appears. I will concentrate on a particular point, the chosen notion of *convergence*, which is to express the success of the regarded process. Mainly two types of convergence concepts seem to be used, we call them *approximative* and *inductive*, respectively.

Consider the simplest, mostly regarded scenario of *supervised* learning, where functions are to be learned from pairs 'argument, function value'. *Approximative* learning means that when more and more data are provided, the corresponding hypotheses will converge in probability to the target function (or a best possible approximation to it): really achieving it only in the limit and only in a probabilistic sense, but, on the other hand, in such a way that the quality of hypotheses at given times can in fact be estimated. In contrast to that, successful *inductive* learning means that the generating of hypotheses will at some time "click into its right place", i.e., become absolutely correct and stay so for ever; but this is paid with the fact that, in general, the learner will never know how good his current hypothesis really is. Approximation is convergence with respect to a specific distance measure between functions, while inductive convergence is convergence w.r.t. the discrete distance measure and hence to any distance measure whatever. Inductive convergence means that in the "discrete" space of a *language*, designed to represent functions, an expression (program) is eventually found which in fact identifies the searched function, while approximative convergence corresponds to a more semantic-oriented search in the function space given.

Why does this mirror the difference between a more traditional – so to say, "static" or "Bourbakian" – kind of mathematics and an algorithmic one? It has been a long, long time that we know that something being true does by far not mean that it could – even in principle – be proved, and it was the algorithmic way of thinking which changed logic at this point. Now for learning (or "induction"), the same kind of difference appears: Should the possessing or achieving of a right concept (a "truth", as we may put it) be considered identical with knowing about this possession? Or should, rather, the possession of a right concept, a right rule to follow – in the sense that one just acts according to it – be regarded as something totally different and far away from the knowing about that possession?

But let us for the moment shift that more principal question to the background and ask a more obvious one. It appears anyway to be a natural desire to combine the benefits of both convergence concepts, i.e., to combine the ultimate finding of a definitely correct program after finitely many observations with the possibility to estimate the quality of any current hypothesis. Our results are from a paper (Menzel and Stephan, 2002), which is to appear in winter 2002/2003. By these results, then, something more from that philosophical question above will come in sight, "implementing" them in a more concrete way.

We aim at bringing the view of statistical learning theory in the world of computable functions, say (in the usual standardization) from $\mathbb{N}$ to $\mathbb{N}$, and investigate to what extent inductive learning can be accompanied with a gradual and assessable improvement of the hypotheses. Our results will heavily rest upon the simple fact that, in contrast to more general situations, any probability distribution over a countable domain has the property that there are always finite sets with arbitrarily high probability less than 1.

A Short Glimpse on Inductive Inference

$\mathbb{N}$ is the set of natural numbers including 0. We assume familiarity with the basic notions from computability theory (see (Odifreddi, 1989), (Soare, 1987)) and probability measures. Small Latin letters $f, g, \ldots$ will be used to denote total functions, small Greek letters $\phi, \psi, \ldots$ for denoting partial ones. $\psi(x) \downarrow$ means that ψ is defined at argument x, $\psi(x) \uparrow$ that it is not. $dom(\psi) = \{x : \psi(x) \downarrow\}$. REC is the class of total recursive functions from $\mathbb{N}$ to $\mathbb{N}$, PREC that

of partial recursive ones. A *numbering* is a two-place partial recursive function ψ, and for any $e \in \mathbb{N}$, ψ_e is the function in PREC with $\psi_e(x) = \psi(e, x)$. ψ *enumerates* the class $\{\psi_e : e \in \mathbb{N}\} \subseteq$ PREC, and any $F \subseteq$ PREC is *recursively enumerable (r.e.)* if it can be enumerated by an appropriate numbering. PREC itself is r.e., REC is not. For a subfamily $F \subseteq$ REC it is a quite restrictive property to be r.e., and these classes play an important rôle in inductive inference, as a particularly good-natured species. Among the numberings enumerating the whole of PREC there are distinguished ones, called *acceptable* or *Gödel* numberings, which correspond to the functional semantics of universal programming languages, where legal programs are identified with their respective numbers in some effective listing. As usual, we will fix some particular Gödel numbering φ as our "standard" basis for computing.

Inductive inference was initiated by Gold (Gold, 1967), see also (Jain et al., 1999), (Odifreddi, 1989), (Osherson et al., 1986). In the framework of computable functions, it conceives learning as finding from finitely many examples a program for an initially unknown function $f \in$ REC, i.e., an e with $\varphi_e = f$. (We will here only deal with the case of total functions as those to be learned.) A *learner* is a machine which reads the graph of an unknown function f in growing finite portions and from time to time, during this process, puts out a number as *hypothesis* for a program of f. As compared to the literature, our following definition is slightly modified in order to fit the later statistical aspects.

Definition 1: *A* **learner** *is a program for a computable partial function*

$$L : (\mathbb{N} \times \mathbb{N})^* \to \mathbb{N} \cup \{?\}$$

where ? is a special extra symbol, and for all sequences $((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathbb{N} \times \mathbb{N})^*$*, if* $L((x_1, y_1), \ldots, (x_m, y_m)) \downarrow$ *then* $L((x_1, y_1), \ldots, (x_k, y_k)) \downarrow$ *for all* $k \leq m$*.*

Here, $L((x_1, y_1), \ldots, (x_m, y_m)) \downarrow = ?$ means that $L((x_1, y_1), \ldots, (x_m, y_m)) \downarrow$ and $L((x_1, y_1), \ldots, (x_m, y_m)) = ?$. Correspondingly, $L((x_1, y_1), \ldots, (x_m, y_m)) \downarrow = e$ for an $e \in \mathbb{N}$. Interpret $L((x_1, y_1), \ldots, (x_m, y_m)) \downarrow = e$ as: The learner having read examples $(x_1, y_1), \ldots, (x_m, y_m)$ puts out e as a *hypothesis* for a program of the unknown function. *The term "hypothesis" will never be used for the symbol ?.* We shortly write $L((x_1, y_1), \ldots, (x_m, y_m)) \downarrow \in \mathbb{N}$ for: There is an $e \in \mathbb{N}$ with $L((x_1, y_1), \ldots, (x_m, y_m)) \downarrow = e$. Read $L((x_1, y_1), \ldots, (x_m, y_m)) \downarrow = ?$ as: The learner has decided not to put out a hypothesis but instead requests to see more data. $L((x_1, y_1), \ldots, (x_m, y_m)) \uparrow$ means that the learner has got stuck in never ending internal computations. For $m = 0$, $((x_1, y_1), \ldots, (x_m, y_m))$ is the empty sequence λ. In our definition, *ambiguous* sequences (where $x_i = x_j, y_i \neq y_j$ for some i, j) are also allowed. These will not be of any importance in our present considerations, so let us agree that all sequences regarded in the following are unambiguous. The learner is *total* iff $L((x_1, y_1), \ldots, (x_m, y_m)) \downarrow$ for all sequences $((x_1, y_1), \ldots, (x_m, y_m))$. *For sake of simplicity, also the function L itself (instead of a program for L) will be called "learner".*

Thus, given an infinite sequence $(x_1, f(x_1)), (x_2, f(x_2)), \ldots$ coming from some f, a learner produces a finite or infinite sequence of hypotheses $e_0, e_1, \ldots$, sometimes remaining tacit for a while (i.e., producing *?*) because of recognized need of more information, and possibly remaining tacit for ever in a non-recognized manner because of never ending computing.

In general, a learner is constructed with respect to a class S of functions: Information about S may be built in L, while the "learning" consists of additionally identifying a particular $f \in S$. Definition 2 below expresses learning in the mode of *syntactical convergence*. Here, an infinite sequence $(x_1, f(x_1)), (x_2, f(x_2)), \ldots$ is called a *valid description* of f if every $x \in \mathbb{N}$ occurs among the x_k.

Definition 2: *(Gold, 1967) A class* $S \subseteq$ *REC is* **Ex-learnable** *(***explanatorily learnable***) iff there is a learner L such that for all* $f \in S$ *and all valid descriptions* $(x_1, f(x_1)), (x_2, f(x_2)), \ldots$ *of f,*

- *there are infinitely many m with* $L((x_1, f(x_1)), \ldots, (x_m, f(x_m))) \downarrow \in \mathbb{N}$*;*

- *and there is an* $e \in \mathbb{N}$ *such that* $\varphi_e = f$ *and* $L((x_1, f(x_1)), \ldots, (x_m, f(x_m))) \downarrow \in \{e, ?\}$ *for almost all* m.

Note that we have *not* demanded that from some "convergence point" onward no more *?* will occur. This is because of our intended combining with approximation, but it fits perfectly well to the idea of inductive inference: As the learner never knows about its possibly reached success, "thinking phases" may well occur in that "stable" period, too. (Anyway it is always possible to dispense with *?* at all by defining $L(\lambda) = 0$ and afterwards simply repeating the last hypothesis until the next one appears; but this amounts to a loss of information unfavourable for our further analysis.)

As has been remarked, REC is not recursively enumerable. But there are rather rich subclasses S of REC which are (e.g., the primitive recursive functions), i.e., there is a total recursive g with $S = \{g_0, g_1, \ldots\}$. Every such class is Ex-learnable:

Example 3 (Learning by Enumeration): *Given an r.e. class of total functions* $g_0, g_1, g_2, \ldots$, *the algorithm* **Learning by Enumeration** *computes for any (unambignous) input* $((x_1, y_1), \ldots, (x_m, y_m))$ *the first index* e *with* $g_e(x_k) = y_k$ *for* $k = 0, \ldots, m$ *and outputs* $h(e)$, *where* h *is a fixed computable translation from* g*-indices to* φ*-indices. Evidently, this algorithm learns any* g_i *from any valid description of* g_i.

There are Ex-learnable classes of total recursive functions which are not contained in any r.e. subclass of REC. A prominent example is the class of *self-describing* functions $S = \{f : f \in \text{REC}, f = \varphi_{f(0)}\}$. A learner for S simply waits (produces *?*) until some example $(0, e)$ shows up and from then onward outputs e. But S is not contained in any r.e. subclass of REC: For any total numbering g there is a self-describing f not in $\{g_0, g_1, \ldots\}$. Define functions f_i by $f_i(0) = i$ and $f_i(x+1) = g_x(x+1) + 1$. By the recursion theorem (Odifreddi, 1989), (Soare, 1987), there is an e with $f_e = \varphi_e$. f_e is self-describing and not contained in $\{g_0, g_1, \ldots\}$.

Bārzdiņš (Bārzdiņš, 1974) generalized Ex-learning to the wider concept of learning in the sense of *semantical convergence*, called *BC-learning* ("behaviourally correct"). L BC-learns f iff any valid description of f causes L to produce infinitely many hypotheses, almost all of which are indices of f. Thus in contrast to Ex-learning, the set of output hypotheses $\{e_0, e_1, \ldots\}$ may be infinite.

Definition 4: *(Bārzdiņš, 1974)* $S \subseteq$ *REC is* **BC-learnable** *(***behaviourally correct***) iff there is a learner* L *such that for all* $f \in S$ *and all valid descriptions* $((x_1, f(x_1)), (x_2, f(x_2)), \ldots)$ *of* f

- *there are infinitely many* m *with* $L((x_1, f(x_1)), \ldots, (x_m, f(x_m))) \downarrow \in \mathbb{N}$*; and*
- *for almost all output hypotheses* e, $\varphi_e = f$.

Clearly, every Ex-learner of some S is also a BC-learner of S. But there are BC-learnable classes which are not Ex-learnable. An example is

$$\{f : f \in \text{ REC}, \varphi_{f(0)} \text{ is a finite variant of } f\},$$

see (Case and Smith, 1983).

Probabilities and Approximation

We will now deal with the statistical approach to learning, which is the second root of our questions. As we are concentrating upon computable functions from $\mathbb{N}$ to $\mathbb{N}$, probability distributions over $\mathbb{N}$ are the natural basis for further analysis. It is common use in this context to apply such a probability distribution pr not only for describing the occurence of data,

but also for establishing a corresponding distance measure between functions (see Vidyasagar (Vidyasagar, 1997) for a discussion of this point). Thus for total functions f, g we define

$$D_{pr}(f,g) = pr(\{x : f(x) \neq g(x)\}).$$

Similarly for partial functions ϕ, ψ:

$$D_{pr}(\phi,\psi) = pr(\{x : \phi(x)\uparrow \text{ or } \psi(x)\uparrow \text{ or } \phi(x)\downarrow \neq \psi(x)\downarrow\}),$$

so that functions are considered to disagree at any place where at least one of them is undefined (see (Menzel and Stephan, 1999) for a more detailed discussion of this view). Furthermore, to pr and a total f we associate a probability distribution on $\mathbb{N}\times\mathbb{N}$ by

$$pr_f(\{x,y\}) = \begin{cases} pr(\{x\}) & \text{if } y = f(x) \\ 0 & \text{otherwise .} \end{cases}$$

Note that, in contrast to "Lebesque-style" probability measures over subsets of $\mathbb{R}$ which are often considered in statistical learning theory (where countable sets have measure zero), pr over $\{0,1\}^{\mathbb{N}}$ has the property that for all $\epsilon > 0$ there is a finite $E \subseteq \mathbb{N}$ with $pr(E) \geq 1-\epsilon$. It will often be appropriate to demand a little more, namely $pr(\{x\}) > 0$ for all $x \in \mathbb{N}$, and probabilities with this property will be called *fair*. The main reason for focusing on fair probabilities is that these have the nice property that for infinite sequences $(x_1, x_2, \ldots)$

$$pr^{\infty}(\{(x_1,x_2,\ldots) : (\forall x)(\exists k)[x = x_k]\}) = 1,$$

where pr^{∞} is the standard extension of pr to infinite sequences, using cylinder sets. Hence given f, an infinite sequence of examples for f will with probability 1 contain every $(x, f(x))$. As a consequence, $D_{pr}(\phi,\psi) = 0$ for partial functions ϕ, ψ if and only if ϕ, ψ are total and equal. Furthermore, on the total functions every metric D_{pr} generated by some fair pr is isometric to the *standard* metric (Menzel and Stephan, 1999), (Menzel and Stephan, 2002), (Odifreddi, 1989) D_{sm} defined as

$$D_{sm}(f,g) = \begin{cases} 0 & \text{if } f = g \\ \frac{1}{min\{x : f(x)\neq g(x)\}+1} & \text{otherwise.} \end{cases}$$

Note that even if pr is fair, it is only with probability 1 that infinite sequences $((x_1, f(x_1)), (x_2, f(x_2)), \ldots)$ generated according to pr_f are valid: Invalid sequences like $((0, f(0)), (0, f(0)), \ldots)$ do exist, although they altogether have measure 0. Such invalid sequences might well contain some relevant information, as is the case for, e.g., $((0, f(0)), (0, f(0)), \ldots)$ when the self-describing functions are to be learned. As a consequence, our notion of successful learning will merely mean success with probability 1.

Learning in statistical learning theory is uniform approximating in probability. It is defined in the PAC understanding ("probably approximately correct") as introduced by Valiant (Valiant, 1984). We will here follow the exposition by Vidyasagar (Vidyasagar, 1997), abstracting in our more principal questions from complexity issues, which are nonetheless quite important for some kinds of applications.

Suppose we know for f, a hypothesis e, and numbers $x_1, \ldots, x_m$ that $\varphi_e(x)\downarrow = f(x)$ for all $x \in \{x_1, \ldots, x_m\}$. Then $D_{pr}(f, \varphi_e) \leq 1 - pr(\{x_1, \ldots, x_m\})$. As in general we will not know pr, this is not of much use. But we may apply the *Chernoff-bound* (see, e.g., (Vidyasagar, 1997)) to approximate $pr(\{x_1, \ldots, x_m\})$: Given $\epsilon, \delta > 0$, there is a number $cher(\epsilon,\delta) = \ulcorner \frac{1}{2\epsilon^2} ln\,(\frac{1}{\delta})\urcorner$ such that *for all* pr, if among $cher(\epsilon,\delta)$ randomly drawn examples h are in $\{x_1, \ldots, x_m\}$ then the probability that $|pr(\{x_1, \ldots, x_m\}) - \frac{h}{cher(\epsilon,\delta)}| < \epsilon$ is at least $1-\delta$. We may use this fact in such a way that a learner (to be constructed) can delay its next hypothesis (i.e., produce *?*) until a sufficiently strong consistency check has been passed with high probability, this way ensuring sufficiently good hypotheses throughout.

To prepare for this technique, we will instead of parameters ϵ, δ above use a single one, n, to express the quality of an approximation; n will also serve to count output hypotheses, and will thus provide a bridging between statistical and inductive behaviour.

For f and some $X \subseteq \mathbb{N}$, let $f \restriction X$ be the restriction of f to X:

$$(f \restriction X)(x) = \begin{cases} f(x) & \text{if } x \in X \\ \uparrow & \text{otherwise} \end{cases}$$

Much of our later results will rest on the following

PROPOSITION 5: *There is a never terminating probabilistic algorithm B, the* **Basic Approximation Algorithm**, *such that the following holds for all probability distributions pr on $\mathbb{N}$ and all $n \in \mathbb{N}$.*

- *There exists a constant $s_{pr,n}$ such that B having seen $s_{pr,n}$ examples according to pr has with probability greater than $\frac{n}{n+1}$ gathered a finite set $X_{pr,n}$ satisfying $pr(X_{pr,n}) > \frac{n}{n+1}$, and has put it out.*

Hence $D_{pr}(f, f \restriction X_{pr,n}) < \frac{1}{n+1}$ with probability greater than $\frac{n}{n+1}$ for all total functions f (not necessarily computable). The sets $X_{pr,n}$ are generated for $n = 0, 1, 2, \ldots$ in order, and it is with probability 1 that they are all eventually put out.

Sketch of Proof: (see (Menzel and Stephan, 2002)). B is the following algorithm.

1: $m = n = 0$, $X = \emptyset$;

2: $i = 2$;

3: $k = 1 + cher(\frac{1}{4(n+1)}, \frac{2^{-i}}{n+1})$, draw k examples $x_{m+1}, \ldots, x_{m+k}$ according to pr,

$h = |\{j : 1 \leq j \leq k,\ x_{m+j} \in X\}|$,

$X = X \cup \{x_{m+1}, \ldots, x_{m+k}\}$;

4: $m = m + k$, $\quad i = i + 1$;

5: if $\frac{h}{k} \leq \frac{2n+1}{2n+2}$ then go to 3;

6: $X_{pr,n} = X$, $\quad$ output $X_{pr,n}$;

7: $n = n + 1$, $\quad$ go to 2.

The idea behind the computation of $X_{pr,n}$ is to postpone its output until enough data $x_1, \ldots, x_m$ have been seen so as to guarantee that $pr(\{x_1, \ldots, x_m\}) > \frac{n}{n+1}$. As pr is unknown, this cannot be checked directly, so the Chernoff bound is used to obtain an estimate for $pr(\{x_1, \ldots, x_m\})$ (steps 3 and 5).

More specifically, one concludes as follows.

(a) With probability 1, all $X_{pr,n}$ are eventually put out.

Proceeding by induction, assume that $X_{pr,n-1}$ has already been produced (the case $n = 0$ is treated in a similar way as what follows). Regard the loop which begins at step 3. As long as it has not terminated, more and more examples are drawn, and at some time the current X will satisfy $pr(X) \geq \frac{4n+3}{4n+4}$. For every i from then on, one easily verifies that the probability that the algorithm has not terminated for this i is at most $\frac{2^{-i}}{n+1}$. Thus the probability that the loop never terminates is zero.

(b) By (a), there is for every n a first a_n such that the probability that B outputs $X_{pr,n}$ for an $i \leq a_n$ is at least $\frac{2n+1}{2n+2}$. Define $s_{pr,n}$ to be the total number of examples drawn when for each $n' \leq n$, i in the inner loop of the algorithm is counted up to $a_{n'}$.

(c) If $X_{pr,n}$ is put out for some i, it is (by the Chernoff bound) with probability at most $\frac{2^{-i}}{n+1}$ that $pr(X_{pr,n}) \leq \frac{n}{n+1}$, hence the overall probability that $X_{pr,n}$ with $pr(X_{pr,n}) \leq \frac{n}{n+1}$ is put out for some $i \leq a_n$ is at most $\sum_{i=2}^{a_n} \frac{2^{-i}}{n+1} < \frac{1}{2n+2}$. Moreover, by definition of a_n, the probability that no output occurs up to $i = a_n$ is at most $\frac{1}{2n+2}$. We have for each n that the probability that for up to a_n runs through the inner loop either no outputs occurs or $X_{pr,n}$ with $pr(X_{pr,n}) \leq \frac{n}{n+1}$, is put out, is less then $\frac{1}{n+1}$. Summing up for all $n' \leq n$ and by definition of $s_{pr,n}$, one concludes that after $s_{pr,n}$ examples drawn, the probability that $X_{pr,n}$ with $pr(X_{pr,n}) > \frac{n}{n+1}$ has been put out is greater than $\frac{n}{n+1}$.

(d) $D_{pr}(f, f \upharpoonright X_{pr,n}) < \frac{1}{n+1}$ with probability greater than $\frac{n}{n+1}$ for all total functions f is immediate.

□

Learners are usual, deterministic algorithms, not, as B is, probabilistic ones. But we may plant a learner into a probabilistic environment: just endow its input sequences with a probability of their occurence. The sequence of its outputs then becomes a sequence of random variables, depending on infinite input sequences. Given the learner L, $n \in \mathbb{N}$, and an infinite sequence $(x_1, y_1), (x_2, y_2), \ldots$, let e_n denote the n'th hypothesis of L on this sequence (outputs *?* are not counted). Because of technical reasons, we begin counting hyptheses with 0 (e_0 may or may not be $L(\lambda)$). Regarding e_n as a random variable, we arrive at the following definition.

Definition 6: *Let S be a class of total functions. The learner L (uniformly)* **approximates** *S iff the following holds for all probability distributions pr on $\mathbb{N}$ and $n \in \mathbb{N}$. There is a constant $s_{pr,n}$ such that for all $f \in S$, when examples $(x_1, f(x_1)), (x_2, f(x_2)), \ldots$ are drawn w.r.t. pr_f and e_n is the n'th hyothesis of L w.r.t. the infinite sequence arising, then $D_{pr}(\varphi_{e_n}, f) < \frac{1}{n+1}$ with probability greater than $\frac{n}{n+1}$. L is a* **universal approximator** *iff L approximates the whole class of all total functions from $\mathbb{N}$ to $\mathbb{N}$, including the non-computable ones.*

Theorem 7: *There exists a universal approximator.*

Sketch of Proof: The learner U when reading examples according to pr_f just simulates B (on their x-parts) and, for $n = 0, 1, \ldots$, outputs its n'th hypothesis u_n if and when B outputs an $X_{pr,n}$ after the finite sequence seen so far. Thus, for $n \in \mathbb{N}$ and (unambiguous) $((x_1, y_1), \ldots, (x_m, y_m))$: If and when B having drawn exactly $x_1, \ldots, x_m$ outputs $X_{pr,n}$, define $U((x_1, y_1), \ldots, (x_m, y_m)) = u_n$ to be a canonical index of the finite function σ_n, where $\sigma_n(x_k) = y_k$ for $k = 1, \ldots, m$ and $\sigma_n(x) \uparrow$ if $x \notin \{x_1, \ldots, x_m\}$; otherwise $U((x_1, y_1), \ldots, (x_m, y_m)) = ?$. □

In Definition 6, it is of course *not* generally the case that $\lim_{n\to\infty} \varphi_{e_n} = f$, in the sense of pointwise convergence, because pr might generate invalid sequences. (Pointwise convergence means that $\lim_{n\to\infty} \psi_n = f$ iff for all x there is an n_x such that for all $n \geq n_x : \psi_n(x) \downarrow = f(x)$.) But remembering our remarks on fairness we have the

Corollary 8: *Let U be a universal approximator, and for any pr, f, n let $e_{pr,f,n}$ be the random variable, which is the n'th output hypothesis of U when examples are drawn according to pr_f (where $e_{pr,f,0}$ counts as the 0'th hypothesis). Then*

$$\lim_{n\to\infty} \varphi_{e_{pr,f,n}} = f \text{ with probability } 1$$

for all fair pr and all total $f : \mathbb{N} \to \mathbb{N}$, in the sense of pointwise convergence.

Remark: The main point in Proposition 5 and Theorem 7 is of course that while $s_{pr,n}$ depends on pr, the algorithms B and U do not. If we had allowed pr to be built in the algorithm then establishing a bound s such that after s examples a set X with $pr(X) > \frac{n}{n+1}$ has with confidence $\frac{n}{n+1}$ been collected would have been fully trivial. This is in some contrast to the exposition in (Vidyasagar, 1997), where the dependency of quality bounds from pr is somehow identified with the algorithm's "knowing" pr (pages 153, 195). In our opinion, such an understanding misses the central point.

A second remark is on Vidyasagar's distinguishing between PAC and PUAC learning, the latter being the stronger notion. In our context, L being PUAC on S for a given pr would mean: For every n, the probability of the set of sequences $(x_1, \ldots, x_m)$ satisfying

$$(\exists f \in S)\left[D_{pr}(\varphi_{L((x_1,f(x_1)),\ldots,(x_m,f(x_m)))}, f) \geq \frac{1}{n+1}\right]$$

goes to zero when m goes to infinity. It is easily verified that our universal approximator in fact has this property, w.r.t. the class of all total functions from $\mathbb{N}$ to $\mathbb{N}$ and any given pr.

Combining inductive learning with approximation

We are now prepared to define our intended "combining" of inductive learning with approximation. In the following definition, we will speak of "types of inductive learning". Examples of such types are Ex, BC, and Num, the latter being learning by enumeration (Example 3), but there are a lot of others in the literature.

Definition 9: *Let S be a class of total functions from $\mathbb{N}$ to $\mathbb{N}$, T a type of inductive learning, and L be a learner.*

L **combines T-learning of S with approximation** if for all fair probabilities pr and $n \in \mathbb{N}$ there is a constant $s_{pr,n}$ such that for all $f \in S$

- L T-learns f with probability 1;
- with probability greater than $\frac{n}{n+1}$: L has put out its n'th hypothesis e_n after at most $s_{pr,n}$ examples read, and $D_{pr}(f, \varphi_{e_n}) < \frac{1}{n+1}$.

L **combines T-learning of S with universal approximation** iff the second requirement is satisfied for all total f (not only those in S).

A first consequence of Theorem 7 refers to dense classes of functions. S is called *dense* if every finite partial function possesses a total extension in S. Examples of dense classes are, e.g., the total functions almost everywhere 0, the polynomials, the primitive recursive functions.

Corollary 10: *Let S be a dense class of total functions, and let the learner L approximate S. Then L can be transformed into a universal approximator M by just skipping the first output hypothesis and afterwards behaving like L. Moreover, if L combines Ex-learning or BC-learning of S with approximation, respectively, then M does so with universal approximation.*

We omit the relatively easy proof, see (Menzel and Stephan, 2002). □

Our next result shows that the rather weak BC-learning and the highly restrictive and simple learning by enumeration can both be combined with universal approximation.

Theorem 11: *Every BC-learnable class of total functions possesses a learner which combines BC-learning with universal approximation. Every class that can be learned by enumeration possesses a learner which combines learning by enumeration with universal approximation.*

Sketch of Proof: (Menzel and Stephan, 2002). The following learner M is constructed from a BC-learner L of S and our universal approximator U. On a given infinite sequence, the hypotheses e_n of M are put out at the same times when U puts out its respective hypotheses u_n, and e_n is defined by patching $\sigma_n = \varphi_{u_n}$ with the last hypothesis a_n which L has produced up to the time where u_n occurs. I.e., e_n ist such that

$$\varphi_{e_n}(x) = \begin{cases} \sigma_n(x) & \text{if } x \in dom(\sigma_n) \\ \varphi_{a_n}(x) & \text{otherwise.} \end{cases}$$

M approximates some f whenever U does, and by the patching of the φ_{a_n} into φ_{e_n}, if L BC-learns f then so does M.

The proof of the second part is similar. L's hypotheses are now indices of total functions, and if these are patched with the σ_n then L being an enumeration learner for S immediately implies that so is M. (In particular, M's convergence is now a syntactical one.) □

The mainly interesting type of inductive convergence is of course that of Ex-learning: It is here where our search for combining two paradigms got its essential motivation. Note that the above construction for BC would not work in the Ex-case, where syntactical convergence is required: Even if L would Ex-learn S, patching the φ_{a_n} with the σ_n might cause infinitely many hypotheses to arise (though almost all of them would belong to the target function f, if $f \in S$). Thus in the above construction, hypotheses are somehow too "flexible" or "irritable", somehow one should prevent them from changing too easily. This suggests the idea of a *consistency* requirement for producing hypotheses: Discard one only if this is enforced by an inconsistency revealed by new data.

Definition 12: *L is* **weakly consistent** *iff for every x there is a number $b_x \geq x$ such that for all unambiguous sequences $((x_1, y_1), \ldots, (x_m, y_m))$ with $\{0, \ldots, b_x\} \subseteq \{x_1, \ldots, x_m\}$, $L((x_1, y_1), \ldots, (x_m, y_m)) \downarrow = e$ where e is such that $\varphi_e(x) \downarrow = y_k$ for any k with $x = x_k$.*

It is easily verified that one can w.l.o.g. assume that a weakly consistent learner is total and never outputs *?*. There are two notions in the literature which are related to weak consistency. To be *globally consistent* ((Jain et al., 1999), Definition 5.61(d)) is equivalent to the special case $b_x = x$ in our definition. Being *reliable* (Minicozzi, 1976) means that the learner either learns or else diverges, i.e., outputs infinitely many hypotheses. (The latter must then, in particular, occur for any non-computable f.) Global consistency is strictly stronger than weak consistency, and being reliable is strictly weaker (Menzel and Stephan, 2002).

Theorem 13: *For a class S of total functions, the following are equivalent properties.*

(1) There is a learner which combines Ex-learning of S with universal approximation.

(2) S is weakly consistently Ex-learnable.

Sketch of Proof: Note that two properties rather alien to each other must be brought in relation: one referring to a statistical environment the learner acts in, and the other one, without such reference, of a more logical nature of "local agreeing".

(1) ⇒ (2). Let L combine the Ex-learning of S with universal approximation. We fix the probability distribution $pr(\{x\}) = \frac{1}{x+1} - \frac{1}{x+2}$ and construct from L a learner M which weakly consistently Ex-learns S. $M((x_1, y_1), \ldots, (x_m, y_m))$ is defined by induction on m. If $m \leq 2$ then $e = M((x_1, y_1), \ldots, (x_m, y_m))$ is any program consistent with $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ (the data being assumed to be unambiguous). Otherwise let e' be $M((x_1, y_1), \ldots, (x_{m-1}, y_{m-1}))$.

1: For all k, n, $3 \leq n \leq k \leq m$, compute the probability of the following event $E_{k,n}$: If k numbers are drawn according to pr then they are all in $\{x_1, \ldots, x_m\}$, and L having seen these numbers together with the corresponding y-values has already put out its n'th hypothesis e_n.

2: Choose the largest n such that $n = 2$ or for some k with $n \leq k \leq m$, $E_{k,n}$ has probability at least $\frac{n-1}{n+1}$.

3:
- If $n > 2$ and the probability of ($E_{k,n}$ *and* $e_n = e'$) is at least $\frac{2}{n+1}$ then $e = e'$;
- else if $n > 2$ and there are e'' such that the probability of ($E_{k,n}$ *and* $e_n = e''$) is at least $\frac{2}{n+1} + 2^{-2-e''}$ then let e be the minimal such e'';
- otherwise let e be any program consistent with $\{(x_1, y_1), \ldots, (x_m, y_m)\}$.

M is weakly consistent: Choose b_x so large that $b_x \geq x + s_{pr,(x+1)(x+2)+1}$ and if one draws $s_{pr,(x+1)(x+2)+1}$ numbers then with probability at least $\frac{(x+1)(x+2)+1}{(x+1)(x+2)+2}$ these are all below b_x. One concludes that the probability of $E_{s_{pr,(x+1)(x+2)+1},(x+1)(x+2)+1}$ is at least $\frac{(x+1)(x+2)+1}{(x+1)(x+2)+2}$. For an unambiguous input $((x_1, y_1), \ldots, (x_m, y_m))$ and (x, y) with $(x, y) = (x_l, y_l)$, then, the n computed in step 2 is at least $(x + 1)(x + 2) + 1$. If e in step 3 is selected according to the first or second case then $e = e_n$ with probability at least $\frac{2}{(x+1)(x+2)+2}$, and it follows by the definitions of $s_{\ldots}$, pr, and D_{pr} that $\varphi_e(x) \downarrow = y$. If e is selected according to the third case then $\varphi_e(x) \downarrow = y$ because of $b_x \geq x$.

M Ex-learns S: Given $f \in S$, let r_i be the probability that L converges to i when data according to pr_f are presented. One shows that there is an i with $r_i > 2^{-i-2}$, and that M must converge to an $e \leq i$. Because of M's weak consistency, e must be an index of f.

(2) $\Rightarrow$ (1). A learner L which combines Ex-learning of S with universal approximation is constructed from a weakly consistent Ex-learner M of S and a universal approximator U. We have to use a slightly sharper version of universal approximation than that one in Definition 6 (Theorem 7 can easily be shown to remain true). Recall from the literature that $\varphi_{e,m}$ is the function with $\varphi_{e,m}(x) = \varphi_e(x)$ if program e on input x halts within m steps, and is undefined otherwise.
$L(\lambda) = ?$. For $m \geq 1$, $L((x_1, y_1), \ldots, (x_m, y_m))$ is computed as follows.

1: Let n be the number of hypotheses put out by L for previous sequences $((x_1, y_1), \ldots, (x_k, y_k)), k < m$, and $c = M((x_1, y_1), \ldots, (x_m, y_m))$. (Recall that M does not produce $?$.) Check whether u_n is known, i.e., $u_n = U((x_1, y_1), \ldots, (x_k, y_k))$ for some $k \leq m$.

2:
- If u_n is known and $c \geq n$ then $L((x_1, y_1), \ldots, (x_m, y_m)) = u_n$;
- else if u_n is known, $c < n$, and $\varphi_{c,m}(x) = \varphi_{u_n}(x)$ for all $x \in dom(\varphi_{u_n})$ then $L((x_1, y_1), \ldots, (x_m, y_m)) = c$;
- otherwise $L((x_1, y_1), \ldots, (x_m, y_m)) = ?$.

For any fair distribution, L Ex-learns every $f \in S$ with probability 1: Let the valid sequence $(x_1, f(x_1)), (x_2, f(x_2)), \ldots$ be given. As M learns f and by the definition of L, there are s and c such that $\varphi_c = f$ and L outputs c or $?$ for all $((x_1, f(x_1), \ldots, (x_m, f(x_m)), m \geq s$. But it will occur infinitely often that $(\forall x \in dom(\varphi_{u_n}))[\varphi_{c,m}(x) \downarrow = \varphi_{u_n}(x)]$ for some u_n produced by U, and hence that $L((x_1, f(x_1)), \ldots, (x_m, f(x_m))) = c$.

L is a universal approximator: Using the mentioned sharper version of U as a universal approximator one can conclude that for all pr, n there is a $t_{pr,n}$ such that after the occurence of $t_{pr,n}$ examples, it is with probability at least $\frac{2n+1}{2n+2}$ that L has produced a hypothesis extending φ_{u_n} and, simultaneously, all of M's later hypotheses will agree with f on $dom(\varphi_{u_n})$. We can now argue as in the proof of Proposition 5 and Theorem 7 to conclude that with probability greater than $\frac{n}{n+1}$, L after $t_{pr,n}$ examples has output its hypothesis e_n and this satisfies $D_{pr}(f, \varphi_{e_n}) < \frac{1}{n+1}$. □

For dense classes, the notions of Ex-learnability combined with universal approximation and of Ex-learnability combined with approximation coincide. A further consequence of Theorem 13 is that in the case of $\{0, 1\}$-valued functions Ex-learnability combined with universal approximation coincides with learnability by enumeration (Menzel and Stephan, 2002). This is

because Ex-learnability by a reliable learner implies enumeration learnability in the $\{0,1\}$-case (Zeugmann, 1983).

The class S of self-describing functions (see above, after Example 3) possesses an Ex-learner combined with approximation (of S!). A slight modification yields an Ex-learnable class for which Ex-learning cannot be combined with approximation, and which may thus serve as a witness to separate both properties (Menzel and Stephan, 2002). It consists of all f such that

- either there is a self-describing g satisfying $(\forall x)[f(x) = g(x) + 1]$,
- or $f(x) = 0$ for almost all x.

Distribution-Free Approximation

Thus far, approximation was such that bounds on the numbers of drawn examples, to ensure a certain quality of approximation, were allowed to depend on the underlying distribution. What happens if we require those bounds to be independent from pr? If such a demand is combined with the uniformity w.r.t. the target function f (as in Definition 6), one arrives at a rather extreme situation. In order to be not too restrictive right away we consider a notion, so to say, "orthogonal" to approximation from Definition 6: Quality bounds are independent from pr, but may depend on the target function f.

Definition 14: *A learner L* **approximates** *a total function f* **in a distribution-free way** *iff for all $n \in \mathbb{N}$ there exists a bound $s_{f,n}$ such that for all distributions pr on $\mathbb{N}$ if examples are read according to pr_f the following holds with probability greater than $\frac{n}{n+1}$: L puts out its n'th hypothesis e_n after at most $s_{f,n}$ examples read, and $D_{pr}(\varphi_{e_n}, f) < \frac{1}{n+1}$. Let T be a type of (inductive) learning and S a class of total functions. L* **combines T-Learning of S with distribution-free approximation** *iff L T-learns S and approximates every $f \in S$ in a distribution-free way. L* **combines T-learning of S with uniform distribution-free approximation** *iff L combines T-learning of S with distribution-free approximation in such a way that the bounds $s_{f,n}$ belonging to the $f \in S$ can be chosen independent from f, i.e., as constants s_n.*

Theorem 13 characterized the possibility to combine Ex-learning with universal approximation by the property "weakly consistent", which does not refer to any statistical context. Our main result on approximation in a distribution-free way is similar, and it is now "proper" (not weak) consistency which comes into play.

Definition 15: *L is* **consistent on** *S iff for all $f \in S$ and sequences $((x_1, f(x_1)), \ldots, (x_m, f(x_m)))$, $L((x_1, f(x_1)), \ldots, (x_m, f(x_m))) \downarrow \in \mathbb{N}$ and $\varphi_{L((x_1,f(x_1)),\ldots,(x_m,f(x_m)))}(x_k) = f(x_k)$ for $k = 1, \ldots, m$. L is* **consistent** *iff L is consistent on the class of all total functions.*

Remark: In inductive inference many learning criteria remain unchanged when only "normalized" sequences of the form $(0, f(0)), (1, f(1)), \ldots$ are considered, so most publications confine themselves to using normalized sequences. The latter is also the case for consistency, though the notion of consistency, in fact, *is* changed by normalization. As confining to normalized sequences would certainly be inappropriate for our purposes, we chose the above definition.

Theorem 16: *If S has a learner that combines Ex-learning with distribution-free approximation then there is an Ex-learner of S which is consistent on S.*

Sketch of Proof: (Menzel and Stephan, 2002). The proof is very similar to part (1) $\Rightarrow$ (2) in the proof of Theorem 13. Observe the difference: The assertion is now much stronger in that at any given time, agreement with the given examples must be ensured "right now". But in order to achieve this, we have now much more possibilities at our disposal, namely, are allowed to

construct an appropriate distribution which might depend on f and even the given sequence. This pr is chosen in such a way that simulating the given L with respect to pr and selecting a hypothesis if it has sufficiently often been put out by L forces the new machine M to be consistent. □

For dense classes S, the converse of Theorem 16 is also true. We have the following characterization.

THEOREM 17: *The following properties of a dense class S of total recursive functions are equivalent*

(1) There is a learner which combines Ex-learning of S with distribution-free approximation

(2) There is a consistent Ex-learner of S

(3) There is a numbering ψ of partial recursive functions such that $S \subseteq \{\psi_e : e \in \mathbb{N}\}$ and the set $\{(e, x, y) : \psi_e(x) \downarrow = y\}$ is recursive.

Observe that we have now obtained the equivalence of a property "in terms of approximative Ex-learning", (1), with a purely recursion-theoretic one, (3), where (2) plays a kind of mediating rôle between both.

Sketch of Proof: (1) ⇒ (2): Immediate, as consistency of a learner on a dense class implies consistency "at all".

(2) ⇒ (3): ψ is based on a one-one recursive enumeration σ_e, $e \in \mathbb{N}$, of all finite unambiguous sequences of pairs of natural numbers. For $\sigma_e = ((x_1, y_1), \ldots, (x_m, y_m))$ define $\psi_e(x) \downarrow = y$ iff

- either $(x, y) \in \{(x_1, y_1), \ldots, (x_m, y_m)\}$
- or $x \notin \{x_1, \ldots, x_m\}$ and $L((x_1, y_1), \ldots, (x_m, y_m), (x, y)) = L((x_1, y_1), \ldots, (x_m, y_m))$

where L is the assumed consistent Ex-learner of S (L must be total).

(3) ⇒ (1): This is a modification of learning by enumeration, using the Chernoff-bound. Given ψ, the learner L computes its n'th hypothesis e_n on a given sequence $(x_1, y_1), (x_2, y_2), \ldots$ in the following way. $e_0 = L(\lambda) = 0$. Assume $e' = e_{n-1}$ has already been produced. Then L, counting upwards from e', regards "tentative hypotheses" e: By means of the Chernoff-bound, each of them is checked for consistency with enough data (L produces *?* while requesting these data). e is put out if the test has gone through. □

Let us at last turn to *uniform* distribution-free approximation. Here, main results in statistical learning theory are connected to the concept of *VC-dimension* in the case of $\{0, 1\}$-valued functions, or *P-dimension* in the general case (Vapnik, 1998), (Vidyasagar, 1997). We look at VC-dimenison only in our special connection of functions on $\mathbb{N}$, though this seems not to be a particularly interesting case in original work on that concept.

DEFINITION 18: *The* **VC-dimension** *(Vapnik-Chervonenkis-dimension) of a class T of total functions from $\mathbb{N}$ to $\{0, 1\}$ is the greatest k satisfying*

- *there is a set X of cardinality k such that every function $\psi : X \to \{0, 1\}$ is extended by some $f \in T$*

if only finitely many k have this property, and is infinite otherwise.

See (Vapnik, 1998), (Vidyasagar, 1997) for the theory around VC-dimension and its generalization to arbitrary functions. For $\{0, 1\}$-valued functions, uniform distribution-free learnability is equivalent to having finite VC-dimension, and for general functions, finiteness of P-dimension is sufficient but not necessary. These results are obtained in a quite general setting, so that they hold true in our special case of functions from $\mathbb{N}$ to $\mathbb{N}$, too.

Having finite VC-dimension is an extremely restrictive property. Besides finite classes S, an example of a fairly "natural" class which has it are the polynomials of a fixed degree. In the reading of statistical learning theory, possessing finite VC-dimension is (for $\{0,1\}$-valued functions) in some sense the strongest and simplest kind of "learnability" (namely, uniform approximability) which a class of functions can have. The following example shows that even this extreme property does not imply inductive learnability, not even in the weak sense of BC-learnability. It shows furthermore that a rather simple transformation of the data can immediately destroy the property of having finite VC-dimension. $\langle \ldots \rangle$ below is a recursive one-one mapping from the set $\mathbb{N}^+$ of all non-empty strings onto $\mathbb{N}$.

EXAMPLE 19: *Let S be the class of all functions $g_f, f \in REC$, where*

$$g_f(x) = \begin{cases} 1 & \text{if } x = \langle f(0), f(1), \ldots, f(k) \rangle \text{ for some } k; \\ 0 & \text{otherwise, that is, there is no such } k. \end{cases}$$

S has VC-dimension 1 but is not BC-learnable.

Sketch of Proof: S has VC-dimension 1: Consider $\{x, y\}$ with $x \neq y$. We have to show that there is some $\sigma : \{x, y\} \to \{0, 1\}$ which is not extended by any $g \in S$ ($\{x, y\}$ is not "shattered by S"). We have $x = \langle a_0, \ldots, a_k \rangle$ for unique numbers $k, a_0, \ldots, a_k$. If $y = \langle a_0, \ldots, a_l \rangle$ for some $l < k$ then $g(x) = 1 \Rightarrow g(y) = 1$ for all $g \in S$; choose $\sigma(x) = 1, \sigma(y) = 0$. If $y = \langle a_0, \ldots, a_l \rangle$ for some $l > k$ and numbers $a_{k+1}, \ldots, a_l$ then $g(y) = 1 \Rightarrow g(x) = 1$ for all $g \in S$; choose $\sigma(x) = 0, \sigma(y) = 1$. If y is of none of these two forms, then $g(x) = 0 \vee g(y) = 0$ for all $g \in S$; choose $\sigma(x) = 1, \sigma(y) = 1$.

It is well known that REC is not BC-learnable (Bārzdiņš, 1974), (Jain et al., 1999), (Odifreddi, 1989), (Osherson et al., 1986). An easy reduction shows that BC-learnability of S would imply BC-learnability of REC: Given $f \in$ REC, compute the arguments for g_f from those for f, apply the (assumed) BC-learner of S and translate produced hypotheses in such a way that e is a program for g_f iff its translation is a program for f. □

Concluding remarks

Statistical learning theory, mainly developed by Vapnik (Vapnik, 1998), offers us a fascinating mathematical world. Clear principles are followed and systematically worked out, guiding concepts discovered, strong and brilliant results obtained. It is beautiful mathematics all around, in the spirit of classical functional analysis.

Might that be *too* beautiful? Doubts begin to arise when one tries to model real-life situations by the proposed means, or to design learning machines for real-life applications, say predict stock values or find a winning strategy for a soccer team. Are learning processes characterized sufficiently well by calculating numbers of needed examples from quality bounds? Do we really need and assume "uniformity" when a certain problem (a class S of functions) is given, i.e., assume the required number of examples to be independent from the searched function? In the group of one of the authors, much and successful work has been done on developing learners for real-life problems (Hörnel and Menzel, 1998), (Menzel, 1998), (Merke and Riedmiller, 2002), (Ragg et al., 2002). This was in the methodology of neural networks and reinforcement learning, thus far away from our more abstract scenario above. Anyway, a central observation in that work has been that a construction getting successful depends extremely on the individual problem given, i.e., the class S. But algorithms described in statistical learning theory tend to be rather "unspecific", without particular knowledge about S being built in. This is different for inductive inference, which offers a lot of constructions of learning machines depending on the specificities of a given class S.

If we try to relate the two worlds considered in this paper, we must regard statistical learning theory over *countable* domains of objects. Admittedly, this is not the typical situation adressed

by statistical learning theory, but its approach and results apply to this case, too, and may be tested on it. Regarding the "learning as uniform approximation" paradigm of statistical learning theory over countable domains, we have then arrived at a highly strange situation:

Either one allows quality bounds (on the numbers of needed examples) to depend on the distribution. Then the whole class of all total functions from $\mathbb{N}$ to $\mathbb{N}$ becomes "learnable" (uniformly approximable). Even stronger, the learning algorithm itself does *not* need to depend from *pr*; only the "environment" it acts in influences convergence. Thus, *nothing specific of "learning" seems to be left at all.*

Or, secondly, one requires those bounds to not depend from distributions. Then the whole concept of "learning", so to say, collapses the other way round: Extremely few classes remain to be learnable in this sense, in the case of $\{0,1\}$-valued functions just those of finite VC-dimension.

Has there a battle been won in favour of one of our two competing powers, in favour of the algorithmic way of thinking? One should be cautious, there are many points in our investigations where interesting modifications of the way taken offer themselves. Moreover, our main goal has not been fighting but the combining of two paradigms of learning. It should anyway be better to make love, not war, – though in so many cases both turn out to be the same thing, after some time.

References

Janis Bārzdiņš. Two theorems on the limiting synthesis of functions. In Latvian State University, editor, *Theory of Algorithms and Programs*, volume I, pages 82–88. 1974. In Russian.

John Case and Carl Smith. Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science*, (25):193–220, 1983.

E. Mark Gold. Language Identification in the Limit. *Information and Control*, (10):447–474, 1967.

Dominik Hörnel and Wolfram Menzel. Learning musical structure and style with neural networks. *Computer Music Journal*, 22(4): 44–62, 1998.

Sanjay Jain, Daniel Osherson, James Royer, and Arun Sharma. *Systems That Learn.* The MIT Press, Cambridge, Massachusetts, 1999. revised edition of (Osherson et al., 1986).

Wolfram Menzel. Problem solving with neural networks. In Ulrich Ratsch et al., editors, *Intelligence and Artificial Intelligence – an Interdisciplinary Debate*, 161–177, Springer, 1998.

Wolfram Menzel and Frank Stephan. Topological Aspects of Numberings. *Mathematical Logic Quarterly*, to appear. See also: Interner Bericht 15, Universität Karlsruhe, Fakultät für Informatik, Karlsruhe, 1999.

Wolfram Menzel and Frank Stephan. Inductive versus approximative learning. In Reimer Kühn et al., editors, *Adaptivity and Learning*. Springer, to appear.

Artur Merke and Martin Riedmiller. Karlsruhe Brainstormers – a reinforcement learning way to robotic soccer II. In A. Birk et al., editors, *RoboCup-2001: Robot Soccer World Cup V*, LNCS 2377, 322–327, 2002.

Eliana Minicozzi. Some natural properties of strong-identification in inductive inference. *Theoretical Computer Science*, (2):345–360, 1976.

Piergiorgio Odifreddi. *Classical Recursion Theory*, volume I and II. North-Holland and Elsevier, Amsterdam, 1989 and 1999.

Daniel Osherson, Michael Stob, and Scott Weinstein. *Systems That Learn. An Introduction to Learning Theory for Cognitive and Computer Scientists.* Bradford – The MIT Press, Cambridge, Massachusetts, 1986.

Thomas Ragg, Wolfram Menzel, Walter Baum, and Michael Wigbers. Bayesian learning for sales rate prediction for thousands of retailers. *Neurocomputing*, 43: 127–144, 2002.

Robert I. Soare. *Recursively Enumerable Sets and Degrees. A Study of Computable Functions and Computably Generated Sets.* Springer, Heidelberg, 1987.

Leslie G. Valiant. A theory of the learnable. *Communications of the Association for Computing Machinery*, 27(11):1134–1142, 1984.

Vladimir N. Vapnik. *Statistical Learning Theory.* John Wiley, New York, 1998.

Mathukumalli Vidyasagar. *A Theory of Learning and Generalization.* Springer, London, 1997.

Thomas Zeugmann. A-posteriori characterizations in inductive inference of recursive functions. *Journal of Information Processing and Cybernetics (EIK)*, 19:559–594, 1983.

On an Algebraic Description of Colorability of Planar Graphs

Michal Mnuk

Institut für Informatik
Technische Universität München
Garching, Germany
E-mail: mnuk@in.tum.de

Abstract

We provide a ideal theoretic formulation of the proof of the famous Four Color Theorem. The connection between graph theory and polynomial ideals yields an alternative view onto the problem of colorability of planar graphs and may contribute to a better understanding of the phenomena arising there.

KEYWORDS: graph colorability, planar graphs, polynomial ideals, Four Color Theorem

1. Introduction

The question of how many colors are needed to properly color vertices of a graph appeared already in the beginnings of the development of graph theory. Around 1850 this question was restricted to planar graphs which ignited a whole spectrum of activities. They culminated in the proof of the Four Color Theorem in 1976 by Apel and Haken ([Apel and Haken, 1976, 1977, 1989]). However, the proof contained a long list of cases which could be verified only by a computer. And even though this proof has been partially simplified ([Robertson et al., 1996, 1997]), hundreds of cases are still left.

Our aim is to introduce algebraic concepts into the problem of colorability of (planar) graphs. While it is not hard to establish a connection between the colorability of general graphs and polynomial ideals, no such correspondence is known for planar graphs. Though, it is not unreasonable that interesting connections will be discovered.

In this paper, we present an ideal theoretic formulation of the proof of the Four Color Theorem. Despite of the complexity of the original proof, we will reformulate it and replace those parts that introduce combinatorial difficulties

by testing polynomial ideal membership. Even though the basic structure of our proof is basically identical to that coming from pure graph theory, the algebraic concepts developed here are by no means mere transcriptions of the graph theoretical ones. For this reason, the theory developed in this paper does not yield (yet) an alternative proof of the Four Color Theorem but only an equivalent formulation of it.

The merits of this approach are not clarified yet. So far, no substantial simplification of the aforementioned proof could be obtained. On the other hand, our approach yields for the first time a representation of some planarity notions by polynomial ideals. This opens a door for a realm of methods from polynomial ideal theory. Moreover, the ideals introduced here are rather regular and are explicitly given. This results in sufficiently special instances of the polynomial ideal membership problem. We hope that understanding the structure of these ideals, e.g. by means of Gröbner bases, will yield new insights and results in colorability of planar graphs, and graphs in general.

2. Graphs, Polynomial Ideals, and Colorability

There are several possibilities to associate polynomials with graphs. This fact is mainly used to encode graph properties either in the values or in the coefficients of the corresponding polynomial. As we do not work with single graphs but with classes of graphs, we are looking for a way to map these classes not to singular polynomials but to polynomial ideals.

In [Mnuk, 2001] the author studied a way to describe properties of graphs by polynomial ideals. There, it turned out that the *graph polynomial* defined below served the intended purpose best.

Definition: Let $G = (V, E)$ be a graph on n vertices $\{1, \ldots, n\}$. With each vertex $i \in V$ we associate a variable x_i. Let $\preceq$ be a connex partial order on the set of variables $\{x_1, \ldots, x_n\}$ (i.e. a reflexive, transitive, and antisymmetric order with $x_i \preceq x_j$ or $x_j \preceq x_i$ for any i, j). The *graph polynomial* f_G of G is given by

$$f_G(x_1, \ldots, x_n) = \prod_{\substack{\{i,j\} \in E \\ x_i \succeq x_j}} (x_i - x_j).$$

Note that the polynomial f_G is homogeneous of degree $|E|$.

Throughout the paper we will denote by k an arbitrary field and by $k[x_1, \ldots, x_n]$ the ring of polynomials over k. Moreover, for simplicity we assume that the variables are ordered $x_1 \preceq x_2 \preceq \ldots \preceq x_n$. At this moment it may not be clear why an ordering on variables is needed – except for eliminating redundant factors from the graph polynomial. Later, when Gröbner bases of ideals will be considered, the ordering will become important.

The graph polynomial can be used to "represent" colorability. A graph $G =$

(V, E) is said to be *colorable by r colors* if there is an assignment of colors to vertices such that no two vertices connected by an edge are assigned the same color.

Let

$$U_r^n := (x_1^r - 1, \ldots, x_n^r - 1). \tag{1}$$

In [Matiyasevich, 1974], Yu. Matiyasevich proved the following theorem.

THEOREM 2.1: *Let $G = (V, E)$ be a graph, f_G its graph polynomial and $n := |V|$. Then G is not colorable with at most r colors if and only if $f_G \in U_r^n$.*

Proof: This theorem has a short proof based on Gröbner bases. We sketch it here.

If $f_G \in U_r^n$, then it is clear that there can be no coloring with at most r colors. On the other hand, assume that G admits no coloring. We fix a degree compatible ordering $\preceq$ on $k[x_1, \ldots, x_n]$. Since $\{x_1^r - 1, \ldots, x_n^r - 1\}$ is a Gröbner basis of U_r^n, we can write

$$f_G = \mathrm{nf}_{U_r^n}(f_G) + g,$$

where $\mathrm{nf}_{U_r^n}(f_G)$ is the normal form of f_G and $g \in U_r^n$. The degree of each variable in $\mathrm{nf}_{U_r^n}(f_G)$ is at most $r - 1$. Evaluating f_G at all r-th roots of unity, we infer from the Fundamental Theorem of Algebra applied to multivariate polynomials that $\mathrm{nf}_{U_r^n}(f_G)$ must vanish identically. □

The above theorem reveals the fact that the connection between graphs and polynomial ideals is by far not straightforward. While it is a trivial observation that an graph that is not r-colorable is also not $(r - 1)$-colorable, the proof of its algebraic analogon is not easy at all.

Thus, expressing theorems from graph theory in algebra is not merely an easy translation. It maps the problems into a similar but different setting from which new insights may be gained.

3. The Four Color Theorem

The history of coloring planar graphs is long and well known. Several books about graph theory contain reports on it (we refer for example to [West, 1996]).

The first proof that every planar graph can be colored by at most five colors was found by Heawood in 1890 ([Heawood, 1890]). It was done in the course of proving the original conjecture which said that four colors suffice for every planar graph. It was finally proved in 1976 by Apel and Haken ([Apel and Haken, 1976, 1977, 1989]). Their proof resulted in a case distinction that, even after it has been simplified, still contains hundreds of cases. It splits into two tasks. First, it is easily shown that there is a finite set of graphs Γ such that every planar graph contains at least one element of Γ as a subgraph. In the more difficult second part it is proved that every planar graph containing an element from Γ as subgraph can be colored by four colors.

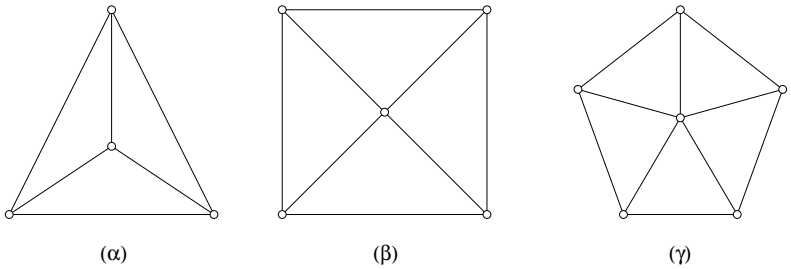


Figure 1: Configurations

Now, we will elaborate on the first part of the proof where a set of graphs is selected that must be contained in every planar graph.

Definition: A *configuration* is a "wheel" with n vertices of degree 3 on the rim and one vertex in the middle (the nut) which is connected to every vertex on the rim. The three smallest configurations denoted α, β and γ are depicted in Figure 1.

Definition: Let C be a set of (planar) graphs. A configuration ω is called *r-reducible* with respect to C if for any graph $G \in C$ containing ω, the following condition holds:

If G without the nut of ω is r-colorable, then G is r-colorable.

Definition: Let $\mathcal{C}$ be a set of graphs. A set U is called *unavoidable for $\mathcal{C}$* if every graph in $\mathcal{C}$ contains some graph from U.

Using Euler's Formula, it is not hard to see that a planar graph $G = (V, E)$ can have at most $3|V| - 6$ edges. From this it easily follows that it must contain a vertex of degree at most 5. Hence, since we assumed that all planar graphs are triangulated, each of them must contain one of the configurations α, β, or γ as a subgraph. Thus we have:

Lemma 3.1: *Let Π be the set of triangulated planar graphs. The set $\Gamma := \{\alpha, \beta, \gamma\}$ is unavoidable for Π.*

This lemma solves the first problem we posed above. We have found a set of graphs that must appear in every planar graph.

Now, we describe the ideas of the proof of the Four Color Theorem.

Theorem 3.1 (The Four Color Theorem): *Every planar graph is 4-colorable.*

Proof (idea): For the sake of simplicity, we will consider only *triangulated* planar graphs, i.e. those graphs containing the maximum number of edges (which is $3n-6$ where n is the number of vertices). It is clear that this restriction makes the colorability problem not easier.

Suppose, the theorem is false. Let $\mathcal{C}$ denote the set of planar graphs that are not 4-colorable and let G be an element of $\mathcal{C}$ with a minimal number of vertices. We know that G must contain one of the configurations α, β, or γ.

Suppose that all these three configurations are 4-reducible. After deleting the nut we obtain a graph G' with less vertices than G and hence 4-colorable. The reducibility of configurations implies 4-colorability of G. A contradiction. □

In order to complete the above proof, it remains to verify the reducibility of configurations α, β, and γ. Note, that the reducibility of α is easy, that of β can be shown by elementary means, and that of γ represents the hardest part of the proof.

At this point we leave the classical graph theoretic proof of the Four Color Theorem. In the rest of the paper we will establish equivalent conditions to the fact that β and γ are 4-reducible. These conditions will be formulated as polynomial ideal membership problems.

4. Polynomial Ideals and Colorability of Planar Graphs

In this section we formulate algebraic conditions which are equivalent to the reducibility of β and γ. At this point, the reasoning used in the classical proof and that used here significantly differ and there is no direct correspondence any more.

The conditions introduced below are interesting for two reasons. Their structure is relatively easy and there is a hope that they can be settled by formal mathematical methods. Furthermore, the paradigms appearing here apply also to general graphs. Hence, it is reasonable that the study of these conditions will shed more light onto the problem "What makes a graph colorable?" and hence on some fundamental questions in colorability of graphs.

In order to keep the reasoning simple, the colors will be, without loss of generality, the fourth roots of unity. These are the common roots of the generators of U_4^n.

4.1. Algebraic Preliminaries

The ideals we will deal with arise from sets of polynomials with prescribed common zeros. Ideals of this type defined over fields will share the property to be *radical* ideals. In the sequel, we assume that all polynomials and ideals are defined over an algebraically closed ground field k.

Definition: An ideal $I \subseteq k[x_1, \ldots, x_n]$ is called a *radical ideal* if it is equal to its

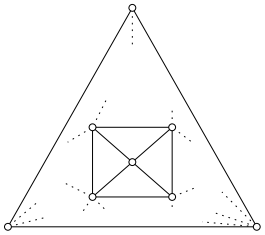


Figure 2: G

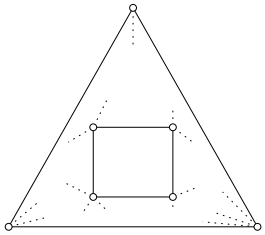


Figure 3: G'

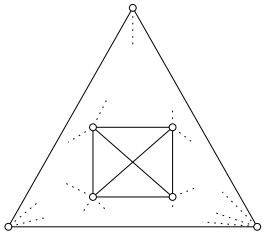


Figure 4: G''

own radical $\sqrt{I}$ defined by

$$\sqrt{I} := \{f \in k[x_1, \ldots, x_n] |\ f^s \in I \text{ for some } s > 0\}.$$

The following theorems collect important properties of radical ideals (details and proofs may be found in many books on commutative algebra, e.g. [Eisenbud, 1994], [Zariski and Samuel, 1958]).

THEOREM 4.1: *Let k be a perfect field and I a zero-dimensional ideal in $k[x_1, \ldots, x_n]$. Then I is a radical ideal if and only if it contains a univariate squarefree polynomial in each variable.*

THEOREM 4.2 (HILBERT NULLSTELLENSATZ): *Let k be an algebraically closed field and $f, g_1, \ldots, g_r$ polynomials from $k[x_1, \ldots, x_n]$. If f vanishes on all common zeros of $g_1, \ldots, g_r$, then there is an integer $s > 0$ such that*

$$f^s \in (g_1, \ldots, g_r).$$

The last theorem has an easy but surprisingly important consequence.

COROLLARY 4.1: *Let $I = (g_1, \ldots, g_r)$ be a radical ideal in $k[x_1, \ldots, x_n]$. If $f \in k[x_1, \ldots, x_n]$ vanishes on all common zeros of $g_1, \ldots, g_r$, then $f \in I$.*

4.2. The Configuration β

After we discussed algebraic preliminaries, we return to the proof of the Four Color Theorem. The essential questions which were left open in the sketch of the proof of Theorem 3.1 pertained to reducibility of configurations β and γ. Now, we will establish a condition that is equivalent to the fact that the configuration β is 4-reducible.

Let us consider a planar (triangulated) graph G containing β (see Figure 2). When the nut together with all adjacent edges is removed, the resulting graph will be denoted G' (see Figure 3). Finally, connecting all vertices on the rim in G' we obtain a new graph G'' depicted in Figure 4.

Now, we assign the nut the variable x_n, the vertices on the rim the variables x_{n-1}, x_{n-2}, x_{n-3}, x_{n-4}, and the rest is assigned arbitrarily. This assignment defines graph polynomials f_G, $f_{G'}$, and $f_{G''}$.

THEOREM 4.3: *The configuration β is 4-reducible if and only if for every planar graph G containing β it holds:*

$$f_{G'} \in (f_{G''}, x_{n-1}^4 - 1, \ldots, x_1^4 - 1) \Longrightarrow f_{G'} \in (x_{n-1}^4 - 1, \ldots, x_1^4 - 1). \qquad (2)$$

Notation: The condition that is equivalent to the 4-reducibility of β will be called "Condition β". The ideal $(f_{G''}, x_{n-1}^4 - 1, \ldots, x_1^4 - 1)$ will be denoted by $I_{G,\beta}$.

Proof: First, we prove the necessity of Condition β. For this, we assume that β is 4-reducible (w.r.t. planar graphs). Let G be a planar graph containing β. The graphs G' and G'' are defined as above.

Case 1. (G' is not 4-colorable) From Theorem 2.1 we obtain $f_{G'} \in (x_{n-1}^4 - 1, \ldots, x_1^4 - 1)$ and the implication (2) holds.

Case 2. (G' is 4-colorable) The reducibility of β implies that G is 4-colorable too. Let $c_1, \ldots, c_{n-1}$ be the colors assigned to vertices $v_1, \ldots, v_{n-1}$. As any 4-coloring of G induces at most 3 colors on the rim of β, the polynomial $f_{G''}$ must vanish. On the other hand, $c_1, \ldots, c_{n-1}$ is a proper 4-coloring and thus $f_{G'}(c_1, \ldots, c_{n-1}) \neq 0$. This shows $f_{G'} \notin (f_{G''}, x_{n-1}^4 - 1, \ldots, x_1^4 - 1)$ making the implication (2) valid.

Hence, the necessity of Condition β is established.

For sufficiency, assume that (2) holds for any planar graph G. To show that the configuration β is 4-reducible, we assume G' to be 4-colorable. If we knew that there is at least one coloring of G' which induces at most three colors on the rim of β, we were done. We prove this assertion by contradiction.

Let us assume that every 4-coloring of G induces four colors on the rim. We choose a perfect algebraically closed field k and consider all common zeros $(c_1, \ldots, c_{n-1})$ of $I_{G,\beta}$ (which are fourth roots of unity). Since $f_{G''}(c_1, \ldots, c_{n-1}) = 0$, there are two possibilities for this to happen. Either there are two vertices "outside" of β that are connected by an edge and that are assigned the same color c_i. Since G' and G'' agree outside of β, we have $f_{G'}(c_1, \ldots, c_{n-1}) = 0$. Or two vertices on the rim of β were assigned the same color. Giving the remaining color to the nut, we would obtain a proper 4-coloring of G with at most three colors on the rim. However, by assumption, no such coloring exists. Hence, a posteriori, there must be an edge on or outside of the rim whose vertices are assigned the same color. Again, we obtain $f_{G'}(c_1, \ldots, c_{n-1}) = 0$.

Summarizing, the polynomial $f_{G'}$ vanishes on all common roots of $I_{G,\beta}$. From Hilbert Nullstellensatz there must be a positive integer s such that $f_{G'}^s \in I_{G,\beta}$. Since $I_{G,\beta}$ is zero dimensional and the ground field is perfect and algebraically closed, Theorem 4.1 implies that it is a radical ideal. Furthermore, Corollary 4.1 shows that $s = 1$ and hence $f_{G'} \in (f_{G''}, x_{n-1}^4 - 1, \ldots, x_1^4 - 1)$. Using Condition β, $f_{G'}$ lies in the ideal $(x_{n-1}^4 - 1, \ldots, x_1^4 - 1)$. The contradiction obtained from Theorem 2.1 shows that there exists at least one 4-coloring which induce at most three colors on the rim. □

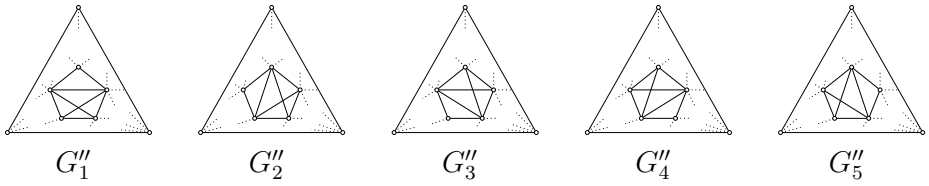


Figure 5:

Remark:

1. The fact that the configuration β is 4-reducible has an elementary graph theoretical proof. We consider bicolored paths joining opposite vertices on the rim of β. The planarity yields then the key argument to show that there is a 4-coloring that induces only three colors on the rim of β.
2. The above proof admits several variations. If the number of colors 4 is replaced by 2 and any reference to the rim of β is replaced by the neighborhood of the nut, the same arguments yield a proof that every graph without odd cycles is bipartite, i.e. can be colored with two colors.

   Replacing 4 by 5 and β by γ we obtain analogous assertions for the Five Color Theorem. Note that the 5-reducibility of α and β is trivial.

4.3. The Configuration γ

The reasoning applied for the configuration β remains the same for γ. Only the ideal will be changed.

Let G be a planar graph containing γ as a subgraph. Analogously to what has been said above, let G' be the graph G where the nut of the pentagon has been removed. Finally, G'' will be replaced by graphs $G''_1, \ldots, G''_5$ (see Figure 5).

THEOREM 4.4: *The configuration γ is 4-reducible if and only if for every planar graph G containing γ it holds:*

$$f_{G'} \in (f_{G''_1}, f_{G''_2}, f_{G''_3}, f_{G''_4}, f_{G''_5}, x_{n-1}^4 - 1, \ldots, x_1^4 - 1) \Longrightarrow f_{G'} \in (x_{n-1}^4 - 1, \ldots, x_1^4 - 1). \quad (3)$$

Notation: The condition that is equivalent to the 4-reducibility of γ will be called "Condition γ". The ideal $(f_{G''_1}, f_{G''_2}, f_{G''_3}, f_{G''_4}, f_{G''_5}, x_{n-1}^4 - 1, \ldots, x_1^4 - 1)$ will be denoted by $I_{G,\gamma}$.

Proof: The arguments showing the necessity of Condition γ can be taken over from the proof of Theorem 4.3. We only have to show that any 4-coloring of G is a common zero of $f_{G''_1}, \ldots, f_{G''_5}$. Such a coloring induces three colors on the rim of γ. Two of them appear twice and a single vertex is colored by the third one. Then it is easy to see that all polynomials $f_{G''_i}$ vanish.

Concerning the sufficiency of Condition γ, let us assume for a planar graph G that G' is 4-colorable. The polynomials $f_{G''_1}, \ldots, f_{G''_5}$ are constructed in such a way that any coloring of G which is their common zero but not a zero of $f_{G'}$ induces exactly three colors on the rim.

Analogously as in the proof of Theorem 4.3, the assumption that there is no coloring inducing three colors on the rim of γ implies that $f_{G'}$ vanishes on all common zeros of $I_{G,\gamma}$. From Hilbert Nullstellensatz for radical ideals and using the validity of Condition γ we infer $f_{G'} \in (x^4_{n-1}-1, \ldots, x^4_1-1)$. The contradiction obtained by Theorem 2.1 concludes the proof. □

5. What Next?

The reformulation of the proof of the Four Color Theorem done in this paper is not just a translation of some theorems into another language. Several concepts in graph theory can be described by polynomial ideals, but the nature of these areas is inherently different. That is why the ideals cannot deliver a one-to-one correspondence and a justification why it makes sense to study such alternative descriptions.

We believe that the study of the ideals introduced in this paper will reveal new aspects of colorability of graphs. Even though the polynomial ideal membership problem is, in general, by far more difficult than the colorability itself, the instances described in this paper are evidently not the most general ones. On the contrary, for a fixed graph G the only polynomials which enter the game arise from G by changing a couple of edges inside of the rim of β and γ.

New insights in this matter may be obtained by studying the structure of the ideals $I_{G,\beta}$ and $I_{G,\gamma}$. Describing their Gröbner bases would be a substantial step toward a solution. Again, computing Gröbner bases is, in general, a demanding task. But both $I_{G,\beta}$ and $I_{G,\gamma}$ arised from the ideal U_4^{n-1} for which a Gröbner basis is known (see (1)).

Another promising direction is the study of the way how a product of two graph polynomials (of planar graphs) reduces with respect to U_4^n. This question touches the base of the theory of colorability of graphs. New insights in this area would also result in a valuable knowledge.

References

K. Apel and W. Haken. Every planar map is four colorable. *Bull. Amer. Math. Soc.*, 82:711–712, 1976.

K. Apel and W. Haken. Every planar map is four colorable. Part I: Discharging. *Illinois J. Math.*, 21:429–490, 1977.

K. Apel and W. Haken. The four color proof suffices. *Math. Intelligencer*, 8: 10–20, 1989.

David Eisenbud. *Commutative Algebra with a View Toward Algebraic Geometry*, volume 150 of *Graduate texts in mathematics*. Springer-Verlag, 1994.

P.J. Heawood. Map-colour theorem. *Q. J. Math.*, 24:332–339, 1890.

Yuri Matiyasevich. A criteria of colorability of vertices of a graph stated in terms of edge orientations. *Diskretnyi Analiz*, 26:65–71, 1974. Institute of Mathematics, Siberian Branch of the Russian Academy of Sciences, (in Russian).

Michal Mnuk. Representing graph properties by polynomial ideals. In V.G. Ganzha, E.W. Mayr, and E.V. Vorozhtsov, editors, *Computer Algebra in Scientific Computing, CASC 2001. Proceedings of the Fourth International Workshop on Computer Algebra in Scientific Computing, Konstanz*, pages 431–444. Springer-Verlag, September 2001.

N. Robertson, D.P. Sanders, P.D. Seymour, and R. Thomas. A new proof of the four color theorem. *Electron. Res. Announc. Amer. Math. Soc.*, 2(1):17–25, 1996.

N. Robertson, D.P. Sanders, P.D. Seymour, and R. Thomas. The four color theorem. *J. Combin. Theory Ser. B.*, 70:2–44, 1997.

Douglas B. West. *Introduction to Graph Theory*. Prentice Hall, 1996.

Oscar Zariski and Pierre Samuel. *Commutative algebra, Volume I*, volume 28 of *Graduate Texts in Mathematics*. Springer-Verlag, 1958.

The Eighth Variation

Teo Mora

DISI, Univ. of Genoa, Genoa, Italy

Abstract

Gröbner bases and related notions, like Macaualay's H-Bases, Hironaka's standard bases, can be described in the setting of graded and filtered rings.

This description required however to apply Buchberger Algorithm and Reduction in a setting in which termination is not available. As a consequence Gröbner representations of elements in a filtered ring P are to be looked for in a series overring $\mathsf{P}^\wedge$, whose constraction and properties are discussed in this paper.

KEYWORDS: Gröbner and standard basis, Buchberger Algorithm

1. Introduction

Already [14] remarked that Buchberger Theory [3, 4, 5, 6] could have been perfectly described and generalized within graded ring theory; my generalization of Buchberger Algorithm to the computation of standard bases of ideals in local rings [15] definitely convinced me that the general setting for describing (commutative and non-commutative) Gröbner bases, Macaualay's H-Bases, Hironaka's standard bases, etc. was graded and filtered rings.

The first attempts [20, 24] to describe Buchberger Algorithm and Theory via filtration theory were however quite disappointing since they were not discussing the natural queries which were forced by the application of Buchberger Reduction in a setting in which termination is not available since the graduation is not necessarily well-ordered:

> Given a filtered ring P, an ideal $\mathsf{I} \subset \mathsf{P}$, an element $h \in \mathsf{I}$ and a Gröbner/standard basis $B := \{b_1, \ldots, b_s\}$ of it
>
> - which kind of Gröbner representations $h = \sum_{i=1}^{s} p_i b_i$ are produced by Buchberger Algorithm in this setting and which other elements $h \in \mathsf{P} \setminus \mathsf{I}$ have the same representation?
> - I.e. what is the description of the overring $\mathsf{P}^\wedge \supset \mathsf{P}$ s.t. $p_i \in \mathsf{P}^\wedge$

- and of the ideal $Cl(\mathsf{I}) \subset \mathsf{P}$ consisting of all elements $h \in \mathsf{P}$ having a Gröbner representation

$$h = \sum_{i=1}^{s} p_i b_i \in \mathsf{P}, p_i \in \mathsf{P}\hat{}?$$

- Does the same theory apply to $\mathsf{P}\hat{}$? □

In fact [24], being mainly interested to a generalization of Buchberger Theory toward the study of subalgebras [23, 2, 22, 21] restricted himself to the case of finite valuation and [20], while in principle aiming to a general formulation, swept the problem under the rug introducing an unjustified assumption which was not even satisfied by the most relevant instances of the theory.

The solution to that queries, once it fulfilled its aim with its direct application to local algebra [16] and to PBW rings [17], was left inside an unpublished draft [18].

This note contains a polished and improved version of that forgotten result* which proves that

- $\mathsf{P}\hat{}$ is the series ring defined over the filtered ring P, in the sense that it is obtained by mimicking Cantor construction of $\mathbb{R}$ as the completion of $\mathbb{Q}$ by adding the topological limits of the converging sequences;
- $Cl(\mathsf{I})$ is the ideal consisting of those elements which are limit of a converging sequence of elements in I;
- B is a Gröbner/standard basis for both I, $Cl(\mathsf{I})$, and $\mathsf{I}\hat{} = \mathsf{I}\mathsf{P}\hat{}$.

More recently the relation between filtration and Buchberger Theory investigated in [23] has been extended to *generalized Gröbner bases filtrations* in [19], and the theories introduced in [27] and [18] have been merged and generalized in [1], giving one of most wide frame available to cover the many generalization of Buchberger Theory.

2. Valuation rings, filtrations and graduations

Let Γ be a (commutative) semigroup, totally ordered (but not necessarily well-ordered) by the semigroup ordering $\succ$, and R be a ring with 1.

A *valuation* is a function† $v : R \mapsto \Gamma$ s.t. $\forall a_1, a_2 \in R \setminus \{0\}$,

- $v(a_1 a_2) = v(a_1) + v(a_2)$;
- $v(a_1 - a_2) \preceq \max(v(a_1), v(a_2))$.

In terms of v we can denote, $\forall \gamma \in \Gamma$:

*The presentation here is limited to the commutative case but the same results can be stated in the non-commutative setting.

†For the notions of this section cf. [28, 8, 25, 7].

Remark that interpretating Buchberger Theory within valuation theory forces to reverse the order of the values in comparison to classical valuation theory.

- $F_\gamma := \{a \in R : v(a) \preceq \gamma\} \cup \{0\} \subset R$,
- $V_\gamma := \{a \in R : v(a) \prec \gamma\} \cup \{0\} \subset R$,
- $G_\gamma := F_\gamma / V_\gamma$.

For each $a \in R \setminus \{0\}$, since we have $a \in F_{v(a)}$ and $a \notin V_{v(a)} \subset F_{v(a)}$ there is necessarily a unique residue class $\mathcal{L}(a) \in G_{v(a)}$ of a mod $V_{v(a)}$, the *initial form* of a.

Let us now denote $G := \bigoplus_{\gamma\in\Gamma} G_\gamma$, the *associated graded ring* of R and $\mathcal{L} : R \mapsto G$ the unique map which associates to each $a \in R \setminus \{0\}$ its initial form $\mathcal{L}(a)$ and which satisfies $\mathcal{L}(0) = 0$.

If we now consider an R-module E and a v-compatible valuation w on it, i.e. a map $w : E \setminus \{0\} \mapsto \Gamma$ s.t. $\forall a \in R \setminus \{0\}, \forall m, m_1, m_2 \in E \setminus \{0\}$,

- $w(am) = v(a) + w(m)$,
- $w(m_1 - m_2) \preceq \max(v(m_1), v(m_2))$,

we can perform the same construction denoting, $\forall \gamma \in \Gamma$:

- $F_\gamma(E) := \{a \in E : w(a) \preceq \gamma\} \cup \{0\} \subset E$,
- $V_\gamma(E) := \{a \in E : w(a) \prec \gamma\} \cup \{0\} \subset E$,
- $G_\gamma(E) := F_\gamma(E)/V_\gamma(E)$,

and denoting $G(E) := \bigoplus_{\gamma\in\Gamma} G_\gamma$ the *associated graded module* of E and $\mathcal{L} : E \mapsto G(E)$ the map s.t. $\mathcal{L}(0) = 0$ and $\forall m \in E \setminus \{0\}, \mathcal{L}(m)$ denotes the unique residue class of m mod $V_{w(a)}(E)$ in $G_{w(a)}(E) \subset G(E)$.

FACT 2.1: *With the notation above it holds,* $\forall a, a_1, a_2 \in R \setminus \{0\}$, $\forall m, m_1, m_2 \in E \setminus \{0\}$,

1. $\forall \gamma \in \Gamma, F_\gamma \subset R$ *is an additive subgroup of* R;
2. $\delta \prec \gamma \Longrightarrow F_\delta \subset F_\gamma, \forall \gamma, \delta \in \Gamma$;
3. $F_\gamma F_\delta \subset F_{\gamma+\delta}, \forall \gamma, \delta \in \Gamma$;
4. *if* $a \neq 0$, *then* $a \in F_{v(a)}$ *and* $a \notin F_\delta$ $\forall \delta \in \Gamma, \delta \prec v(a)$;
5. $\{0\} = \bigcap_{\gamma\in\Gamma} F_\gamma$;
6. *the associated graded ring* G *is a* Γ*-graded ring;*
7. $v(a_1a_2) = v(a_1) + v(a_2), \mathcal{L}(a_1a_2) = \mathcal{L}(a_1)\mathcal{L}(a_2)$;
8. $v(a_1 - a_2) \preceq \max(v(a_1), v(a_2))$;
9. $v(a_1 - a_2) \prec \max(v(a_1), v(a_2)) \iff \mathcal{L}(a_1) = \mathcal{L}(a_2)$;
10. $\mathcal{L}(a_1 - a_2) = \mathcal{L}(a_1) - \mathcal{L}(a_2) \iff v(a_1 - a_2) = v(a_1) = v(a_2)$;
11. $\mathcal{L}(a_1 - a_2) = \mathcal{L}(a_1) \iff v(a_1 - a_2) = v(a_1) \succ v(a_2)$;
12. $\mathcal{L}(a) = 0 \iff a = 0$;
13. $v(1) = 0, \mathcal{L}(1) = 1$;

14. $\forall \gamma \in \Gamma, F_\gamma(E) \subset E$ *is an additive subgroup of* R*;*

15. $\delta \prec \gamma \Longrightarrow F_\delta(E) \subset F_\gamma(E), \forall \gamma, \delta \in \Gamma$;

16. $F_\gamma F_\delta(E) \subset F_{\gamma+\delta}(E), \forall \gamma, \delta \in \Gamma$;

17. if $m \neq 0$*, then* $m \in F_{w(m)}(E)$ *and* $m \notin F_\delta(E)\ \forall \delta \in \Gamma, \delta \prec w(m)$;

18. $\{0\} = \bigcap_{\gamma \in \Gamma} F_\gamma(E)$;

19. the associated graded module $G(E)$ *is a* Γ*-graded* G*-module;*

20. $w(am) = v(a) + w(m), \mathcal{L}(am) = \mathcal{L}(a)\mathcal{L}(m)$;

21. $w(m_1 - m_2) \preceq \max(w(m_1), w(m_2))$;

22. $w(m_1 - m_2) \prec \max(w(m_1), w(m_2)) \iff \mathcal{L}(m_1) = \mathcal{L}(m_2)$;

23. $\mathcal{L}(m_1 - m_2) = \mathcal{L}(m_1) - \mathcal{L}(m_2) \iff w(m_1 - m_2) = w(m_1) = w(m_2)$;

24. $\mathcal{L}(m_1 - m_2) = \mathcal{L}(m_1) \iff w(m_1 - m_2) = w(m_1) \succ w(m_2)$;

25. $\mathcal{L}(m) = 0 \iff m = 0$. □

Example 2.1: Let us present some "classical" examples:

A $\Gamma := \mathbb{N}$, ordered so that $d \prec d+1, \forall d$, $R := k[X_1, \ldots, X_n]$ and for each polynomial $f \in R$ we define $v(f) := \deg(f)$; then we have $G = R$ and if $f = \sum_{i=0}^{d} f_i$, f_i homogeneous of degree $i, \forall i$, and $f_d \neq 0$ then [12, 13] $\mathcal{L}(f) = H(f) := f_d$.

B $\Gamma := \mathbb{N}$, ordered so that $d \succ d+1, \forall d$, $R := k[X_1, \ldots, X_n]$ and for each polynomial

$$f = \sum_{i=0}^{\infty} f_i \in k[X_1, \ldots, X_n] \subset k[[X_1, \ldots, X_n]],$$

f_i homogeneous of degree $i, \forall i$, we define $v(f)$ to be its *order*, i.e. $v(f) := \min\{i : f_i \neq 0\}$; then we have $G = k[[X_1, \ldots, X_n]]$ and [10] $\mathcal{L}(f) = in(f) := f_{v(f)}$;

C $\Gamma := \mathbb{N}$ ordered so that $d \succ d+1, \forall d$, $R := \mathbb{Z}$, $p \in \mathbb{N}$ a prime, and $\forall \nu \in \mathbb{Z}$, $v(\nu) := \max\{n : p^n \mid \nu\}$; then G is the Hensel ring [9] $G \cong \mathbb{Z}_p[[X]]$.

D Generalizing the last two examples, we can consider $\Gamma := \mathbb{N}$ ordered so that $d \succ d+1, \forall d$, a ring R, an ideal $L \subset R$ s.t. $\cap_d L^d = \{0\}$ and for each $a \in R \setminus \{0\}$ we set $v(a) := \min\{i : a \in L^i\}$; then we have:

$$F_d = L^d, V_d = L^{d+1}, G_d = L^d/L^{d+1},$$

and

$$G = \sum_d L^d/L^{d+1}, \mathcal{L}(a) := a \bmod L^{v(a)+1}.$$

E If we now consider

$$\Gamma := \{X_1^{a_1} \cdots X_n^{a_n} : (a_1, \ldots, a_n) \in \mathbb{N}^n\},$$

ordered by $\prec$ and $R := k[X_1, \ldots, X_n]$, then each polynomial $f \in R$ has a unique ordered representation as an ordered linear combination of terms:

$$f = \sum_{i=1}^{s} c_i t_i : c_i \in k \setminus 0, t_i \in \Gamma, t_1 \succ \cdots \succ t_s.$$

If we set $v(f) := t_1$ then $\mathcal{L}(f) = c_1 t_1$ is its *maximal monomial*.

If $\prec$ is well-ordered — $1 \prec X_i, \forall i$ — we are within Buchberger Theory [3, 4, 5] and we have $G = R$.

F If, instead $1 \succ X_i, \forall i$, then $G = k[[X_1, \ldots, X_n]]$ and we are within Hironaka Theory [10].

G In the same setting it is also possible to describe the non-commutative *solvable polynomial rings*, also known as *PBW-rings* [11, 17] whose vectorspace support is $k[X_1, \ldots, X_n]$ endowed with a twist multiplication s.t.

$$X_j X_i = c_{ij} X_i X_j - p_{ij}, 1 \leq i < j$$

where

- $c_{ij} \in k$,
- for no $i, j, k, 1 \leq i < j < k \leq n$, $c_{ij} = c_{jk} = 0$,
- $\forall i, j, 1 \leq i < j \leq n, p_{ij} = \sum_{k=1}^{s} d_k t_k, d_k \in k \setminus 0, t_k \in \Gamma$ satisfies $t_k \prec X_i X_j, \forall k$. □

3. Leitideale and Buchberger reduction

Let, as before

- Γ be a semigroup, totally ordered by the semigroup ordering $\succ$,
- R be a ring with 1,
- $v : R \mapsto \Gamma$ a valuation,
- E an R-module and
- $w : E \setminus \{0\} \mapsto \Gamma$ a v-compatible valuation.

For any set $B \subset E$ we denote $\mathcal{L}\{B\} := \{\mathcal{L}(b) : b \in B\}$ and $\mathcal{L}(B) \subset G$ the submodule generated by B.

For an ideal $\mathsf{I} \subset R$ the ideal $\mathcal{L}(\mathsf{I})$ generated by $\mathcal{L}\{\mathsf{I}\} = \{\mathcal{L}(r) : r \in \mathsf{I}\}$ is called the *leitideal* of I [8]

The Eighth Variation

If $\mathsf{E} \subset E$ is a submodule and we consider the R-module[‡] $N := E/\mathsf{E}$ with the filtration inherited by the one in E, by setting

$$F_\gamma(N) = \frac{\mathsf{E} + F_\gamma(E)}{\mathsf{E}} \text{ and } V_\gamma(N) = \frac{\mathsf{E} + V_\gamma(E)}{\mathsf{E}},$$

one has

$$G_\gamma(N) = G_\gamma(E)/\mathsf{N}_\gamma \text{ where } \mathsf{N}_\gamma := \{\mathcal{L}(m) : m \in \mathsf{E}, w(m) \preceq \gamma\}$$

so that if we denote

$$\mathsf{N} := \bigoplus_{\gamma \in \mathbb{N}} \mathsf{N}_\gamma \subset \bigoplus_{\gamma \in \mathbb{N}} G_\gamma(E) = G(E),$$

we have

$$\mathsf{N} = \bigoplus_\gamma \{\mathcal{L}(m) : m \in \mathsf{E}, w(m) \preceq \gamma\} = (\mathcal{L}(m) : m \in \mathsf{E}) = \mathcal{L}(\mathsf{E})$$

so that

$$G(E/\mathsf{E}) = \bigoplus_\gamma G_\gamma(N) \simeq \bigoplus_\gamma G_\gamma(E)/\mathsf{N}_\gamma \simeq G(E)/\mathsf{N} = G(E)/\mathcal{L}(\mathsf{E}).$$

Therefore the knowledge of the leitmodul $\mathcal{L}(\mathsf{E})$ of E allows to obtain the associated graded module of the quotient module $G(E/\mathsf{E})$.

For a submodule $\mathsf{E} \subset E$, a set $B \subset \mathsf{E}$ is called a *Gröbner basis* or *standard basis* of E if $\mathcal{L}\{B\} = \{\mathcal{L}(b) : b \in B\}$ generates $\mathcal{L}(\mathsf{E})$.

For each $h \in E$ a representation

$$h = \sum_i r_i b_i : r_i \in R, b_i \in B$$

is called a *standard representation* in R in terms of B iff

$$w(h) \geq v(r_i) + w(b_i), \forall i.$$

With reference to the" classical" examples presented in Ex. 2.1 a set $B \subset \mathsf{E}$ s.t. $\mathcal{L}\{B\}$ generates $\mathcal{L}(\mathsf{E})$ is known as

A H-basis,

E Gröbner basis,

F standard basis.

[‡]The most significant case is when $E = R$ so that E is an ideal and N is a quotient ring.

Let us assume that, given a homogeneous element $m \in G(E)$ and a homogeneous set $B = \{b_1, \ldots, b_s\} \subset G(E)$, it is possible to verify whether $m \in (B)$ in which case it is possible to produce homogeneous elements $a_i \in G(R)$ s.t.

$$m = \sum_{i=1}^{s} a_i b_i \text{ and } \deg(m) = \deg(a_i) + \deg(b_i), \forall i.$$

Then, if Γ is well-ordered by $\prec$, Buchberger reduction allows, given an element $h \in E$ and a Gröbner basis $B = \{b_1, \ldots, b_s\} \subset \mathsf{E}$ of E, to decide whether $h \in \mathsf{E}$ in which case it produces elements $p_i \in R$ s.t.

$$h = \sum_{i=1}^{s} p_i b_i \text{ and } w(h) \succ v(p_i) + w(b_i), \forall i.$$

The inductive construction consists in checker whether

- $\mathcal{L}(h) \notin \mathcal{L}(\mathsf{E})$ in which case, by definition, $h \notin \mathsf{E}$, or
- $\mathcal{L}(h) \in \mathcal{L}(\mathsf{E})$ in which case we obtain elements $a_i \in G(R)$ s.t.

$$\mathcal{L}(h) = \sum_{i=1}^{s} a_i \mathcal{L}(b_i) \text{ and } w(h) = \deg(a_i) + w(b_i), \forall i.$$

In the latter case, if we choose any elements $r_i \in R$ s.t. $\mathcal{L}(r_i) = a_i$ and we define

$$h' := h - \sum_{i=1}^{s} r_i b_i,$$

we have $\mathcal{L}(h) = \sum_{i=1}^{s} \mathcal{L}(r_i b_i)$ and

$$w(h) = \deg(a_i) + w(b_i) = v(r_i) + w(b_i) = w(r_i b_i), \forall i,$$

so that $w(h') \prec w(h), w(h') \neq w(h)$.

Then, inductively, $h \in \mathsf{E} \iff h' \in \mathsf{E}$.

This inductive argument is sufficient to prove

Theorem 3.1 (Buchberger): *With the notation above, and under the assumption that Γ is well-ordered by $\prec$, for each $h \in E$ there is a* normal form *$NF(h) := h' \in E$ s.t.*

- *$h' - h$ has a standard representation in R in terms of B, and*
- *$h' \neq 0 \iff \mathcal{L}(h') \notin \mathcal{L}(B)$.*

□

Corollary 3.2 (Buchberger): *With the notation above and under the assumption that Γ is well-ordered by $\prec$, then the following conditions are equivalent:*

1. *B is a standard basis of E;*

2. *for each* $h \in E$, $h \in \mathsf{E}$ *iff it has a standard representation in* R *in terms of* B*;*
3. *for each* $h \in E$ *either*
   - $h \in \mathsf{E}$ *and* h *has a standard representation in* R *in terms of* B*, or*
   - $h \notin \mathsf{E}$ *and there is* $h' \in E \setminus \{0\} : \mathcal{L}(h') \notin \mathcal{L}(\mathsf{E})$ *and* $h - h'$ *has a standard representation in* R *in terms of* B
4. *for each* $h \in E$ *there is* $h' \in E$ *s.t.*
   - $h' - h$ *has a standard representation in* R *in terms of* B*, and*
   - $h' \neq 0 \Longrightarrow \mathcal{L}(h') \notin \mathcal{L}(\mathsf{E}), h \notin \mathsf{E}$,

and all imply that B *is a basis of* E. □

If however, Γ is not well-ordered by $\prec$ in principle the recursive argument could be performed infinitely many times; the properties of the partial results can be in any case described within the following

Lemma 3.3: *Under the notation above, let* $B := \{b_1, \ldots, b_s\} \in \mathsf{E}$ *and* $h \in E$ *and let us recursively define the following sequences*

$$\{p_{ni} : n \in \mathbb{N}\} \subset R, \forall i, 1 \leq i \leq s, \quad \{f_n : n \in \mathbb{N}\} \subset E, \quad \{h_i : n \in \mathbb{N}\} \subset E$$

as follows

- $f_0 := h, p_{0i} := 0, h_0 := 0$;
- *if* $f_j = 0$ *or* $\mathcal{L}(f_j) \notin \mathcal{L}(B)$ *then*

$$f_{j+1} := f_j, \quad p_{j+1\ i} := p_{ji}, \quad h_{j+1} := h_j;$$

- *if* $f_j \neq 0$ *and* $\mathcal{L}(f_j) \in \mathcal{L}(B)$*, and* $r_{ji} \in R$ *are elements s.t.* $\mathcal{L}(f_j) = \sum_i \mathcal{L}(r_{ji})\mathcal{L}(g_i)$ *and* $w(f_j) = v(r_{ji}) + w(b_i), \forall i$*, then*

$$f_{j+1} := f_j - \sum_i r_{ji} b_i, \quad p_{j+1\ i} := p_{ji} + r_{ji}, \quad h_{j+1} := h_j + \sum_i r_{ji} b_i.$$

Then

1. $\forall j, f_j = 0 \Longrightarrow f_{j+1} = 0$;
2. $\forall j, f_j \neq 0, \mathcal{L}(f_j) \notin \mathcal{L}(B) \Longrightarrow f_{j+1} = f_j$;
3. $\forall j, f_j \neq 0, \mathcal{L}(f_j) \in \mathcal{L}(B) \Longrightarrow w(f_{j+1}) < w(f_j) = w\left(\sum_i r_{ji} b_i\right)$;
4. $\forall j, f_j + h_j = h$;
5. $\forall j, h_j \in (b_1, \ldots, b_s) \subset \mathsf{E}$;
6. $\forall j, h_j = \sum_i p_{ji} b_i$ *is a standard representation in* R *in terms of* B. □

In order to understand how much can be generalize Buchberger Corollary 3.2 let us introduce helpfull definitions.

Let Γ be a semigroup, totally ordered by $<$. Γ is said to be *inf-limited* if $\forall \gamma \in \Gamma$ and for each decreasing sequence $\gamma_1 > \gamma_2 > \ldots > \gamma_j > \ldots$ there is n s.t. $\gamma_n \leq \gamma$.

Under the notation above, if $B := \{b_1, \ldots, b_s\} \in \mathsf{E}$ and $h \in E$ a representation $h = \sum_i p_i b_i + h' : p_i \in R, h' \in E$ is called

- a *standard representation* in R in terms of B iff

$$h' = 0 \text{ and } w(h) \succeq v(p_i) + w(b_i);$$

- a *truncated standard representation at* $\gamma \in \Gamma$ in terms of B iff

$$w(h) \succeq v(p_i) + w(b_i) \text{ and } h' \neq 0 \Longrightarrow w(h') \prec \gamma.$$

An element $h \in E$ is said to have a *Cauchy standard representation* in terms of B if, $\forall \gamma \in \Gamma$, it has a truncated standard representation at γ in terms of B.

We can now reformulate Buchberger Corollary as

PROPOSITION 3.4: *Under the notation above, let* $\mathsf{E} \subset E$ *be a submodule of* E *and* $B := \{b_1, \ldots, b_s\} \in \mathsf{E}$.

If Γ *is inf-limited, then the following conditions are equivalent*

1. B *is a standard basis of* E.
2. *each element* $h \in \mathsf{E}$ *has a Cauchy standard representation in terms of* B*;*
3. *for each* $h \in E$ *either*
   - h *has a Cauchy standard representation in* R *in terms of* B*, or*
   - *there is* $h' \in E \setminus \{0\} : \mathcal{L}(h') \notin \mathcal{L}(\mathsf{E})$ *and* $h - h'$ *has a standard representation in* R *in terms of* B

□

Clearly Prop. 3.4 generalizes only the trivial part of Cor. 3.2; in fact, without assuming that Γ is well-ordered Th. 3.1 doesn't hold because we don't have termination.

The rôle of Th. 3.1 and of termination within Buchberger Theory is twofold:

1. it allows to characterize the elements in E as the ones having a standard representation in terms of the standard basis B;
2. the production of normal forms allows, within Buchberger Algorithm, to test whether a basis B is a Gröbner one and, if this is not the case, to enlarge it; in fact Buchberger Algorithm produces a specific finite set of elements $\mathbf{S}(B) \subset \mathsf{E}$, the "S-pairs of B" s.t.
   - B is a Gröbner basis iff $NF(\sigma) = 0, \forall \sigma \in \mathbf{S}(B)$;
   - and if this is not the case, setting $B' := B \cup \{NF(\sigma) : \sigma \in \mathbf{S}(B)\}$ it holds

$$\mathcal{L}(B) \subset \mathcal{L}(B') \subset \mathcal{L}(\mathsf{E}), \mathcal{L}(B) \neq \mathcal{L}(B').$$

It is therefore clear that in order to generalize Cor. 3.2 to those filtration rings where termination doesn't hold, we need to solve the two problems above.

A partial solution of the first one is quite easy to produce: the *closure* of E,

$$Cl(\mathsf{E}) := \bigcap_{\gamma\in\Gamma} \mathsf{E} + F_\gamma(E),$$

satisfies the properties:

- $Cl(\mathsf{E})$ is an R-module;
- if $h \in E$ has a Cauchy standard representation in terms of B, then $h \in Cl(\mathsf{E})$;
- the following conditions are equivalent
  - B is a standard basis of E
  - $\forall h \in E$, $h \in Cl(\mathsf{E})$ iff it has a Cauchy standard representation in terms of B.

This however is not sufficient to solve the crucial problem of characterizing normal forms, which are the essential tool for generalizing Buchberger Algorithm; however Ex. 2.1.**C**, pointing an obvious link between Hensel's and Buchberger's constructions, provides the hint we need for solving this task: the elements $h \in E \setminus \mathsf{E}$ which have a Cauchy standard representation in terms of B can be characterized, using the notation of Lemma 3.3, as those for which $\lim h_n = 0$ so that $\lim f_n = h$ and their standard representation is $h = \sum_i p_i b_i$ where $p_i := \lim p_{ni}, \forall i$.

This can be proved by mimicking Cantor construction (cf. [26]) of the real closure of an ordered field by means of Cauchy sequences.

4. Cauchy sequences

Let us begin by remarking that, for each $\gamma \in \Gamma$,

- $\exists \gamma', \gamma'' \in \Gamma : \forall f \in F_{\gamma'}(E), g \in F_{\gamma''}(E), f + g \in F_\gamma(E)$,
- $\exists \gamma', \gamma'' \in \Gamma : \forall f \in F_{\gamma'}, g \in F_{\gamma''}(E), fg \in F_\gamma(E)$,

and that

$$\bigcap_{\gamma\in\Gamma} F_\gamma(E) = \{0\},$$

and by recalling that

- a sequence $(a_n), a_i \in E, n \in \mathbb{N}$ is called a *Cauchy sequence* in E if

$$\forall \gamma \in \Gamma, \exists n \in \mathbb{N} : a_p - a_q \in F_\gamma(E), \forall p, q > n,$$

- and a Cauchy sequence (a_n) in E is called a *null sequence* if

$$\forall \gamma \in \Gamma, \exists n \in \mathbb{N} : a_p \in F_\gamma(E), \forall p > n,$$

and that the following results hold:

- for each Cauchy sequences (m_n) in E, $\exists \gamma \in \Gamma, n \in \mathbb{N} : w(m_p) \prec \gamma, \forall p > n$;
- the set $\mathfrak{C}(E)$ of all Cauchy sequences in E is an R-module under the operations

$$(m_n) + (\mu_n) := (m_n + \mu_n), a(m_n) := (am_n), \forall (m_n), (\mu_n) \in \mathfrak{C}(E), a \in R;$$

- the set $\mathfrak{C}(R)$ of all Cauchy sequences in R is a ring under the operation

$$(a_n) \cdot (b_n) := (a_n \cdot b_n), \forall (a_n), (b_n) \in \mathfrak{C}(R);$$

- the set $\mathfrak{C}(E)$ of all Cauchy sequences in E is a $\mathfrak{C}(R)$-module under the operation

$$(a_n) \cdot (m_n) := (a_n \cdot m_n), \forall (a_n) \in \mathfrak{C}(R), (m_n) \in \mathfrak{C}(E);$$

- the set $\mathfrak{N}(E)$ of all null sequences in E is a $\mathfrak{C}(R)$-module;
- $\hat{R} := \mathfrak{C}(R)/\mathfrak{N}(R)$ is a ring;
- the map $\phi : R \mapsto \hat{R}$ which associates to each $a \in R$ the residue class $\mathrm{mod}\mathfrak{N}(R)$ of the Cauchy sequence $(a_n) : a_n = a, \forall n$, is an immersion;
- $\mathfrak{N}(R) \cdot \mathfrak{C}(E) \subset \mathfrak{N}(E)$;
- $\hat{E} := \mathfrak{C}(E)/\mathfrak{N}(E)$ is an $\hat{R}$-module;
- the map $\phi : E \mapsto \hat{E}$ which associates to each $m \in E$ the residue class $\mathrm{mod}\mathfrak{N}(E)$ of the Cauchy sequence $(m_n) : m_n = m, \forall n$, is an immersion.

5. Standard bases in valuation rings

If $\mathsf{m} \in \hat{E}$ and (m_n) is a Cauchy sequences in E, we say that (m_n) *converges* to m, $\lim m_n = \mathsf{m}$ if (m_n) belongs to the residue class $\mathrm{mod}\,\mathfrak{N}(E)$ represented by m.

If $\mathsf{m} \in \hat{E}$ and $(m_n), (\mu_n)$ are Cauchy sequences in R which converge to m, then:

- $\exists N \in \mathbb{N} : w(m_p) = w(m_N), \mathcal{L}(m_p) = \mathcal{L}(m_N), \forall p > N$;
- $\exists N' \in \mathbb{N} : w(m_p) = w(\mu_q) =: w\hat{}(\mathsf{m}), \mathcal{L}(m_p) = \mathcal{L}(\mu_q) =: \mathcal{L}\hat{}(\mathsf{m}), \forall p, q > N'$.

This defines maps $w\hat{} : \hat{E} \mapsto \Gamma$, $\mathcal{L}\hat{} : \hat{E} \mapsto G(E)$ and (if we apply the same result to the module $E := R$) $v\hat{} : \hat{R} \mapsto \Gamma$ which satisfie

- $v\hat{}$ is a valuation on $\hat{R}$ which extends the valuation v in R;
- $w\hat{}$ is a $v\hat{}$-compatible valuation on $\hat{E}$, which extends the valuation w in E;
- $\mathcal{L}\hat{}$ extends $\mathcal{L}$;
- $v\hat{}, w\hat{}, G, G(\cdot), \mathcal{L}\hat{}$ satisfy the properties listed in Fact 2.1.
- In particular $G(\hat{R}) = G(R) = G, G(\hat{E}) = G(E)$,

- and, $\forall s, G(\hat{R}^s) \cong G(R^s) \cong G^s$.
- Moreover, in this context the result about $Cl(\mathsf{E})$ can be reinterpreted as

$$\hat{\mathsf{E}} \cap E = Cl(\mathsf{E}) = \bigcap_{\gamma \in \Gamma} \mathsf{E} + F_\gamma(E).$$

If we perform the same construction as described in Lemma 3.3 but starting with an element $h \in \hat{E}$ we obtain sequences

$$\{p_{ni} : n \in \mathbb{N}\} \subset R, \forall i, 1 \leq i \leq s, \{f_n : n \in \mathbb{N}\} \subset \hat{E}, \{h_i : n \in \mathbb{N}\} \subset \mathsf{E}$$

which statisfies the same properties as in Lemma 3.3 and moreover, if we denote $\gamma_n := w(f_n), \forall n$, it holds:

- if $h \in E$ then $\forall n, f_n \in E$;
- the sequence $\gamma_0 \succ \gamma_1 \succ \cdots \succ \gamma_n \cdots$ is an infinite decreasing sequence;
- (f_n) is a Cauchy sequence converging to 0;
- (h_n) is a Cauchy sequence converging in E to h;
- $\forall i, (p_{ni})$ is a Cauchy sequence in R, whose limits in $\hat{R}$ we will denote p_i.

We are therefore now able to state Buchberger Theorem in a valation ring:

THEOREM 5.1: *Let Γ be a (commutative) semigroup, inf-limited by $\prec$, R be a ring with 1, $v : R \mapsto \Gamma$ a valuation, E be an R-module and $w : E \mapsto \Gamma$ be a v-compatible valuation, $\mathsf{E} \subset E$ be a submodule of E and $B := \{b_1, \ldots, b_s\} \in \mathsf{E}$.*

With the notations introduced in this and in the previous sections, then the following conditions are equivalent:

1. *B is a standard basis of E.*
2. *B is a standard basis of $Cl(\mathsf{E})$.*
3. *B is a standard basis of $\hat{\mathsf{E}}$.*
4. *for each element $h \in E$, $h \in Cl(\mathsf{E})$ iff it has a Cauchy standard representation in R in terms of B;*
5. *for each element $h \in \hat{E}$, $h \in \hat{\mathsf{E}}$ iff it has a Cauchy standard representation in R in terms of B;*
6. *for each element $h \in E$, $h \in Cl(\mathsf{E})$ iff it has a standard representation in $\hat{R}$ in terms of B;*
7. *for each element $h \in \hat{E}$, $h \in \hat{\mathsf{E}}$ iff it has a standard representation in $\hat{R}$ in terms of B;*
8. *for each element $h \in E$, $h \in Cl(\mathsf{E})$ iff there is a Cauchy sequence $(h_n) \in \mathfrak{C}(\mathsf{E})$ converging to h and s.t., $\forall n \in \mathbb{N}, h_n$ has a standard representation in $\hat{R}$ in terms of B;*

9. *for each element* $h \in \hat{E}$, $h \in \hat{\mathsf{E}}$ *iff there is a Cauchy sequence* $(h_n) \in \mathfrak{C}(\hat{\mathsf{E}})$ *converging to* h *and s.t.,* $\forall n \in \mathbb{N}, h_n$ *has a standard representation in* $\hat{R}$ *in terms of* B*;*
10. *for each* $h \in E \setminus \{0\}$ *either*
    - $h \in Cl(\mathsf{E})$ *and* h *has a standard representation in* $\hat{R}$ *in terms of* B*, or*
    - $h \notin Cl(\mathsf{E})$ *and there is* $h' \in E \setminus \{0\} : \mathcal{L}(h') \notin \mathcal{L}(\mathsf{E})$ *and* $h - h' \in \mathsf{E}$ *has a standard representation in* R *in terms of* B
11. *for each* $h \in \hat{E} \setminus \{0\}$ *either*
    - $h \in \hat{\mathsf{E}}$ *and* h *has a standard representation in* $\hat{R}$ *in terms of* B*, or*
    - $h \notin \hat{\mathsf{E}}$ *and there is* $h' \in \hat{E} \setminus \{0\} : \hat{\mathcal{L}}(h') \notin \hat{\mathcal{L}}(\hat{\mathsf{E}})$ *and* $h - h' \in \mathsf{E}$ *has a standard representation in* R *in terms of* B*;*
12. *for each* $h \in E \setminus \{0\}$ *either*
    - $h \in Cl(\mathsf{E})$ *and there is a Cauchy sequence* $(h_n) \in \mathfrak{C}(\mathsf{E})$ *converging to* h *and s.t.,* $\forall n \in \mathbb{N}, h_n$ *has a standard representation in* $\hat{R}$ *in terms of* B*, or*
    - $h \notin Cl(\mathsf{E})$ *and there is* $h' \in E \setminus \{0\} : \mathcal{L}(h') \notin \mathcal{L}(\mathsf{E})$ *and* $h - h' \in \mathsf{E}$ *has a standard representation in* R *in terms of* B*;*
13. *for each* $h \in \hat{E} \setminus \{0\}$ *either*
    - $h \in \hat{\mathsf{E}}$ *and there is a Cauchy sequence* $(h_n) \in \mathfrak{C}(\hat{\mathsf{E}})$ *converging to* h *and s.t.,* $\forall n \in \mathbb{N}, h_n$ *has a standard representation in* $\hat{R}$ *in terms of* B*, or*
    - $h \notin \hat{\mathsf{E}}$ *and there is* $h' \in \hat{E} \setminus \{0\} : \hat{\mathcal{L}}(h') \notin \hat{\mathcal{L}}(\hat{\mathsf{E}})$ *and* $h - h' \in \mathsf{E}$ *has a standard representation in* R *in terms of* B*;*

and all imply that B *is a basis of* $Cl(\mathsf{E})$ *in* $\hat{R}$*.*

References

[1] J. Apel, *Computational Ideal Theory in Finitely Generated Extension Rings.* Theor. Comp. Sci. **244** (2000), 1–33.

[2] B. Barkee *Groebner bases. The ancient secret mystic power of the algucompubraicus. A revelation whose simplicity will make ladies swoon and grown men cry.* Report (1988)

[3] B. Buchberger, *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal.* Ph. D. Thesis, Innsbruck (1965).

[4] B. Buchberger, *Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleischunssystem.* Aeq. Math. **4** (1970) 374–383.

[5] B. Buchberger, *Introduction to Gröbner Bases.* In [6] 3–31.

[6] B. Buchberger, F. Winkler *Gröbner Bases and Application*. Cambridge Univ. Press (1998).

[7] D. Eisenbud, *Commutative Algebra with a view toward Alebraic Geometry*. Springer (1995).

[8] W. Gröbner, *Algebraische Geometrie*. Bibliographisches Institut Mannheim (1968).

[9] K. Hensel, *Zahlentheorie*, Goschen (1913).

[10] H. Hironaka, *Resolution of singularities of an algebraic variety over a field of characteristic zero*. Ann. Math. **79** (1964), 197–326.

[11] A. Kandry-Rody, V. Weispfenning, *Non commutative Groebner base in algebras of solving type*. J. Symb. Comp. **9** (1990), 1–26.

[12] F. S. Macaulay, *On the Resolution of a given Modular System into Primary Systems including some Properties of Hilbert Numbers*, Math. Ann. **74** (1913) 66–121

[13] F. S. Macaulay, *The Algebraic Theory of Modular Systems*, Cambridge Univ. Press (1916)

[14] H.M. Möller, F. Mora, *New constructive methods in classical ideal theory*. J.Alg.**100** (1986), 138-178

[15] F. Mora, *An algorithmic approach to local rings*. L.N.C.S. **204** (1985), 518–525.

[16] T. Mora, *La queste del saint* $\mathrm{Gr}_a(A_L)$. *A computational approach to local algebra*. Disc. Appl. Math. **33** (1991) 161–190

[17] T. Mora, *Gröbner bases in non-commutative algebras*, L.N.C.S. **358** (1989), 150–161

[18] T. Mora, *Seven variations on standard bases*. Preprint (1988)

[19] E. Mostaig, M. Sweedler, *Valuations and Filtrations*. J. Symb. Comp., to appear

[20] L. Robbiano, *On the theory of graded structures*. J. Symb. Comp. **2** (1986), 139–170.

[21] L. Robbiano, M. Sweedler, *Subalgebra bases*. L.N. Math. **1430** (1990), 61–87.

[22] D. Shannon, M. Sweedler, *Using Gröbner bases to determine algebra membership, split surjective algebra homomorphisms and determine birational equivalence*. J. Symb. Comp. **6** (1988), 267–273.

[23] D.A. Spear, *A constructive approach to commutative ring theory* in*Proc. of the 1977 MACSYMA Users' Conference*, (NASA CP-2012) (1977) 369–376.

[24] M. Sweedler, *Ideal bases and valuation rings*. Preprint (1986).

[25] W. Vasconcelos, *Computational Methods in Commutative Algebra and Algebraic Geometry*, Springer (1998).

[26] B.L. van der Waerden, *Modern Algebra* Vol. I, Ungar (1949)

[27] G. Zacharias, *Generalized Gröbner Bases in Comutative Polynomial Rings*, Bachelor Thesis, MIT, Boston (1978)

[28] O. Zariski, P. Samuel, *Commutative Algebra* Vol. II, Van Nostrand, Princeton (1960).

Variable Shape Logicographic Symbols

KOJI NAKAGAWA

RISC-Linz, Johannes Kepler University, A-4040 Linz, Austria
E-mail: koji.nakagawa@risc.uni-linz.ac.at

Abstract

The idea of logicographic symbols is to dispatch graphical drawings to predicate or function constants. The drawings symbolize the intuition behind the notion of the constants. Without losing its rigor logicographic symbols can be used in the formal statements whose readability is remarkably enhanced.

In this paper we propose new type of logicographic symbols, called 'variable shape', which change their shape depending on the arguments whereas others, called 'fixed shape', do not change their shape. Also the design principles which should be took into consideration are discussed.

KEYWORDS: inventing new notation, intuitive proof presentation, user interface for integrated mathematical systems

1. Introduction

The computer-support for doing mathematics has significantly increased in the past few decades. The fact can be seen by the successful achievements of several existing computer algebra systems and theorem proving systems. In these systems mathematical knowledge is described in more formal and precise ways than those of mathematical books so that it can be processed in computers. However the problem is that it is often very hard to understand the intuition behind the formal description in such systems. Traditionally in order to help the understanding, we usually make some pictures or drawings. We definitely need tools to reconcile these two issues, formal rigor and intuitive understanding helped by drawings.

As a solution of this problem Buchberger proposed a new idea of logicographic symbols in a paper [Buc2000]. The main idea of logicographic symbols is to dispatch graphical drawings, which have slots for arguments, to predicate or function constants. The

graphical drawings symbolize the intuition behind the notion. These graphically represented drawings can be treated as meaningful objects. Namely it can be used in formal statements and treated completely same as the traditional formal statements. Additionally these can be not only visible but also manipulatable, e.g by changing arguments by modifying the places called slots.

In order to show the feasibility of logicographic symbols we implemented the tool on the top of the Theorema system which is an integrated mathematical environment for proving, computing, and solving [BDK+1997, BDJ+2000]. So far we have shown several examples of logicographic symbols in several conferences [Buc2000, Buc2001, NB2001a, NB2001b] and my Ph.D. thesis[Nak2002].

The purpose of this paper is to propose a new type of logicographic symbols called 'variable shape' which change their shapes depending on the argument terms. On the contrary the existing type of logicographic symbols are called 'fixed shape' because their shapes are independent of the argument terms.

At first we review the 'fixed shape' logicographic symbols, and secondly see the idea of 'variable shape' logicographic symbols by some examples. Finally we discuss some design principles which should be took into consideration.

2. Fixed Shape Logicographic Symbols

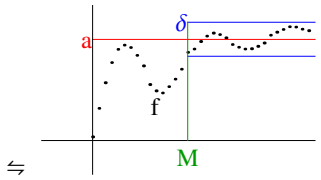
2.1 Example: Limit

In Theorema the definition of 'sequence f comes closer to a than ϵ from N on' can be described by the following declaration labeled by "4-ary limit" where 'any[f,a,ϵ,N]' indicates that 'f,a,ϵ,N' are variables.

$$\textbf{Definition}\Big["4-\text{ary limit}", \text{any}[f, a, \epsilon, N], \text{limit}[f, a, \epsilon, N] : \Longleftrightarrow \underset{\substack{n \\ n \geq N}}{\forall} |f[n] - a| < \epsilon\Big]$$

Using textual constants for denoting functions and predicates, it is sometimes hard to understand the intuition behind the formalized statements. Proofs of propositions based on this notion will of course rely exclusively on the formal definition. On the other hand a drawing conveying the intuitive idea behind this notion will greatly help in understanding propositions and proofs about the notion. In [Buc2000] Buchberger introduced a logicographic symbol for the predicate constant 'limit' of the formula 'limit[f,a,δ,M]' by the following declaration:

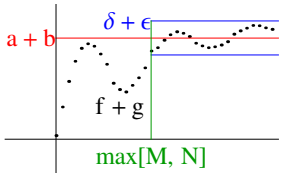
LogicographicNotation["4−ary limit", any[f, a, δ, M],
limit[f, a, δ, M] (f " stays closer to " a " than " δ " from " M)



⇋];

In the above declaration, the entire drawing with four *slots* for the four possible arguments constitutes the new symbol. The label "4-ary limit" can be used to refer to this logicographic definition afterwards. The expression 'any[f, a, δ, N]' means that 'f, a, δ, N' are slots in the declaration. The annotation '(f " stays closer to " a " than " δ " from " M)' indicates a suggestion for reading the formula.

After the activation of the declaration, formulae with 4 arguments predicate constant 'limit' are shown by the right-hand of the declaration in which all slots are replaced by the arguments. For example, 'limit[f+g, a+b, δ+ϵ, max[M,N]]' is shown as follows:



Formulae represented with logicographical symbols can be manipulated just like any other logical formulae of Theorema. Namely, they can be evaluated and their slots can be modified by selecting places of the variable slots and typing the terms.

2.2 Example: Merge Sort

Figure 1 shows a theory for showing the correct of merge sort. The expressions '⟨⟩', '⟨x, $\overline{X}$⟩', 'x ⌣ X', 'X ≍ Y' stand for 'empty tuple', 'a tuple with the first element x and a finite sequence '$\overline{X}$' of elements', 'tuple X with element x prepended', 'concatenation of X and Y', respectively. And 'stmg', 'mg', 'istv', 'ist', 'ipm', 'lsp', 'rsp' stand for 'sorted by merging', 'merged', 'is sorted version of', 'is sorted', 'is permuted version of', 'left split', and 'right split', respectively.

Variable Shape Logicographic Symbols

Algorithm["stmg", any[X],

$$\text{stmg}[X] := \begin{cases} X & \Leftarrow\ |X| \le 1 \\ \text{mg}[\text{stmg}[\text{lsp}[X]], \text{stmg}[\text{rsp}[X]]] & \Leftarrow\ \text{otherwise} \end{cases}];$$

Algorithm["mg" , any[X, Y, a, b, $\bar{x}$, $\bar{y}$],

$$\begin{aligned} &\text{mg}[\langle\rangle, Y] := Y \\ &\text{mg}[X, \langle\rangle] := X \\ &\text{mg}[\langle a, \bar{x}\rangle, \langle b, \bar{y}\rangle] := \\ &\quad \begin{cases} a \frown \text{mg}[\langle \bar{x}\rangle, \langle b, \bar{y}\rangle] & \Leftarrow\ a \ge b \\ b \frown \text{mg}[\langle a, \bar{x}\rangle, \langle \bar{y}\rangle] & \Leftarrow\ \text{otherwise} \end{cases} \end{aligned}];$$

Definition["istv", any[X, Y],
istv[X, Y] $\Longleftrightarrow$ (ist[X] $\wedge$ ipm[X, Y])];

Lemma["mg" , any[A, B],
(ist[A] $\wedge$ ist[B]) $\Longrightarrow$ ist[mg[A, B]]];

Lemma["mg2", any[A, B],
ipm[mg[A, B], A $\asymp$ B]];

Figure 1: Formalized Merge-Sort Theory

With the several infix notations and the case notation of Theorema, the declaration is much more comprehensive than other programming languages. However no notations are dispatched for the newly introduced constants 'stmg', 'mg', etc. We now introduce logicographic symbols for the newly introduced constants. With the facility of logicographic symbols the user has complete freedom in designing new symbols for the various notions. The following may be a possible choice:


mg[X, Y] the result of merging two tuples X and Y


lsp[X] left split of X


stmg[X] the result of sorting X by merging


rsp[X] right split of X


ipm[X, Y] X is a permuted version of Y

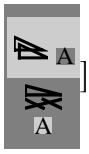

ist[X] X is sorted


istv[X, Y] X is a sorted version of Y

With these logicographic symbols, the above knowledge base can be described in the way shown in Figure 2. Here the expressions are represented in a nested 2-dimensional syntax with dark gray and light gray coloring for clarifying the syntactical structure.

Then the correctness of merge sort can be formalized as follows:

Proposition["mgs" (∗ correctness of merge−sort ∗), any[A],


]

The expected Theorema proof can be found in [NB2001a, Nak2002].

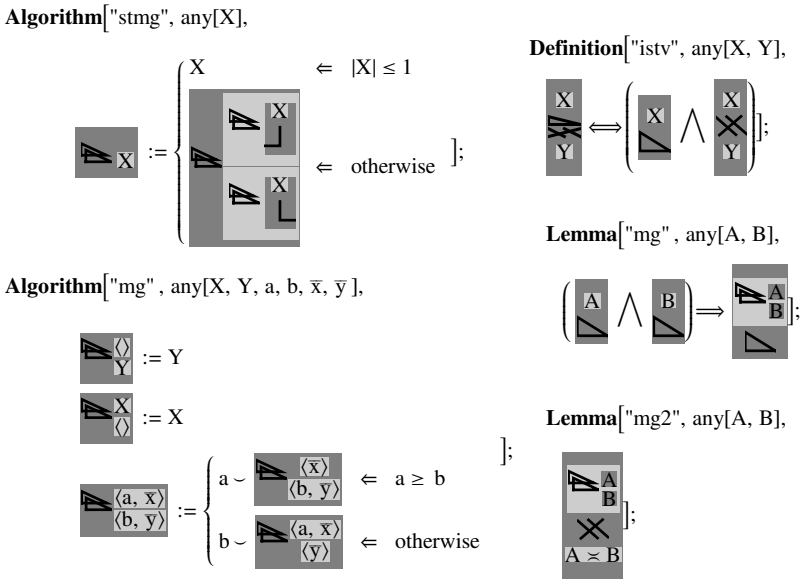


Figure 2: Formalized Merge-Sort Theory with Logicographic Symbols

2.3 Example: Notation of Function in Set Theory

In [Buc2001] Buchberger proposed a technique to compose a new logicographic symbol from existing logicographic symbols by using the example of the notion of function in set theory. Here are the definitions of the notion of function and their logicographic representation:

Definitions["relations", any[A, r, B],

isrel[A, r, B] $\Longleftrightarrow (r \subseteq A \times B)$
isltot[A, r, B] $\Longleftrightarrow \forall_{a \in A} \exists_{b \in B} \langle a, b\rangle \in r$
isrtot[A, r, B] $\Longleftrightarrow \forall_{b \in B} \exists_{a \in A} \langle a, b\rangle \in r$
isrfun[A, r, B] $\Longleftrightarrow \forall_{b1 \in B, b2 \in B} \forall_{a \in A} (\langle a, b1\rangle \in r \wedge \langle a, b2\rangle \in r \Longrightarrow b1 == b2)$]
islfun[A, r, B] $\Longleftrightarrow \forall_{a1 \in A, a2 \in A} \forall_{b \in B} (\langle a1, b\rangle \in r \wedge \langle a2, b\rangle \in r \Longrightarrow a1 == a2)$

LogicographicNotation["relations", any[A, r, B],

isrel[A, r, B] (r "is a relation between " A "and" B) ⇋ A —r— B
isltot[A, r, B] (r "is left total on " A "and" B) ⇋ A |r B
isrtot[A, r, B] (r "is right total on " A "and" B) ⇋ A r| B]
isrfun[A, r, B] (r "is right functional on " A "and" B) ⇋ A r≻ B
islfun[A, r, B] (r "is left functional on " A "and" B) ⇋ A ≺r B

Variable Shape Logicographic Symbols

The notion of being a function can now be expressed by the formula 'isrel[A,r,B]∧isrfun[-A,r,B]' and with the logicographic declaration it is represented as '$A \xrightarrow{r} B \wedge A \overset{r}{>} B$'. However, it is natural to represent this formula in more compact form '$A \xrightarrow{r} B$'.

In order to introduce this representation, we could have an extra 'LogicographicNotation' declaration and its definition. However in this way, in order to introduce all such combined symbols, we would have to introduce 2^5–6(=26) additional logicographic symbols. Instead, at first we introduce a facility to combine existing logicographic symbols by introducing a syntactic construct '∧' (wedge) which is different from the logical construct '∧' (and), and second we dispatch appropriate logicographic symbols for it.

For example, if we have the following expression '(isrel∧isrfun)[A,r,B]', then the logicographic representation of the expression are composed from the representation of 'isrel' and 'isfun' and becomes '$A \xrightarrow{r} B$'.

Introducing a composing logicographic symbol, e.g. 'isrel∧isrfun', means introducing a new predicate or function constant whose logicographic representation can be automatically composed by combining the logicographic representations of its constituents. Thus, in a Theorema session, these expressions can be treated as predicate or function constants. In fact, the meaning could be defined by just adding definitions of the following kind to the Theorema knowledge base:

Definition["functionality", any[A, r, B],
(isrel ∧ isrfun)[A, r, B] ⟺ isrel[A, r, B] ∧ isrfun[A, r, B]]

or alternatively Theorema provers expand the formulae to the appropriate one as the need arises.

Additionally we introduce a new convention. When argument slots in a logicographic symbol are left out, they are considered as quantified argument variables. For example, if the first and third slots are omitted from '$A \xrightarrow{r} B$', we will have the representation '$\xrightarrow{r}$' which is considered as '$\exists_A \exists_B\ A \xrightarrow{r} B$', namely '$\exists_A \exists_B$ (isrel ∧ isrfun)[A, r, B]'.

Under this conventions the theorem states that 'the composition of two bijective functions is bijective' can be described as follows:

Theorem["Composition of Bijective Functions", any[f, g],
$\left(\overset{f}{\leftrightarrow} \wedge \overset{g}{\leftrightarrow} \right) \Longrightarrow \overset{f \circ g}{\leftrightarrow}$]

With 3 lemmata the proof is automatically produced by Theorema as follows:

Prove:

(Theorem (Composition of Bijective Functions))

$\underset{f,g}{\forall}\big(f \leftrightarrow \wedge\ g \leftrightarrow \Rightarrow\ f\circ g \leftrightarrow \big)$,

under the assumptions:

(Lemma (Composition of Injective Functions)) $\underset{f,g}{\forall}\big(f \leftrightarrow \wedge\ g \leftrightarrow \Rightarrow\ f\circ g \leftrightarrow \big)$,

(Lemma (Existence of Bijective)) $\underset{f}{\forall}\big(f \leftrightarrow \Rightarrow |f \overset{f}{\leftrightarrow} f|\big)$,

(Lemma (Bijective is Injective)) $\underset{f}{\forall}\big(f \leftrightarrow \Rightarrow \overset{f}{\leftrightarrow}\big)$,

For proving (Theorem (Composition of Bijective Functions)) we take all variables arbitrary but fixed and prove:

(1) $f_0 \leftrightarrow \wedge\ g_0 \leftrightarrow \Rightarrow\ f_0\circ g_0 \leftrightarrow$.

We prove (1) by the deduction rule.

We assume

(2) $f_0 \leftrightarrow \wedge\ g_0 \leftrightarrow$

and show

(3) $f_0\circ g_0 \leftrightarrow$.

From (2.2), by (Lemma (Bijective is Injective)), we obtain:

(10) $g_0 \leftrightarrow$.

From (2.1), by (Lemma (Bijective is Injective)), we obtain:

(9) $f_0 \leftrightarrow$.

From (9) and (10), by (Lemma (Composition of Injective Functions)), we obtain:

(16) $f_0\circ g_0 \leftrightarrow$.

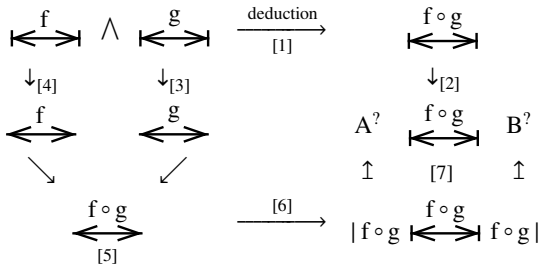
From (16), by (Lemma (Existence of Bijective)), we obtain:

(26) $|f_0\circ g_0 \overset{f_0\circ g_0}{\leftrightarrow} f_0\circ g_0|$.

Formula (3) is proved because (26) is an witness for it.

□

The flow of the produced proof can be seen more legibly by using the following diagrammatic proof presentation. [1] We assume the left hand side and prove the right hand side. [2] In order to prove the goal, we have to find appropriate A and B. [3,4] *Optically* we can grasp the facts by extraction. [5,6] By the lemmata. [7] We found appropriate values for A and B. Therefore the theorem is proved. Producing such diagrammatic proof presentation automatically is an interesting future work.



3. Variable Shape Logicographic Symbols

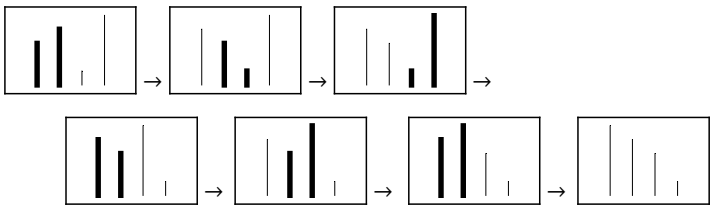
3.1 Example: List Representation

In the previous section we saw some logicographic symbols whose sizes may change, but shape did not change. Namely the shape depends on only the predicate or function constant, but does not depend on the arguments. Here we introduce some logicographic symbols which change their shape depending on the arguments, namely 'variable shape' logicographic symbols.

In mathematics, for heuristic and pedagogic reasons, one often introduces or illustrates abstract notions by accompanying concrete examples. For example, when explaining the bubble-sort algorithm, one would illustrate the effect of the algorithm by showing its trace in concrete examples using lists of numbers. Preferably, one would illustrate the effect by using lists of strokes with varying length so that the effect is more easily visible. For example, when we have the following trace:

$$\langle \mathbf{3}, \mathbf{4}, 1, 5\rangle \to \langle 4, \mathbf{3}, \mathbf{1}, 5\rangle \to \langle 4, 3, \mathbf{1}, \mathbf{5}\rangle \to$$
$$\langle \mathbf{4}, \mathbf{3}, 5, 1\rangle \to \langle 4, \mathbf{3}, \mathbf{5}, 1\rangle \to \langle \mathbf{4}, \mathbf{5}, 3, 1\rangle \to \langle 5, 4, 3, 1\rangle$$

the corresponding visualized trace is more illustrative than the trace above:



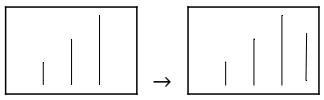
Here lengths of vertical lines vary depending on other elements such that the largest element fits into the box. Also widths between lines vary depending on the number of elements. Note that this is only possible for terms whose ingredient values are known. For terms whose subterms contain variable this is of course not possible.

Variable shape logicographic symbols can not be declared as easily LogicographicNotation declaration used in the examples of the fixed shape logicographic symbols. Instead there should be a mechanism to declare an algorithm to construct the visual representation. For the moment, this functionality has not yet been implemented, but hard-corded in Mathematica.

• Interaction

So far what we could interact with logicographic symbols was to fill in or change the expressions of slots. For this logicographic symbol, different type of interactions is possible. The possible interactions are adding, deleting, arranging, enlarging or shortening bars by mouse operations. These interactions change the elements of lists, e.g. adding an element to the list.

For example, adding an element '2' to the end of the list ⟨1,2,3⟩ can be achieved by drawing a bar into the logicographic symbol:



Note that the system automatically recognizes the size of the element from the vertical length of the bar.

• Tricky Order

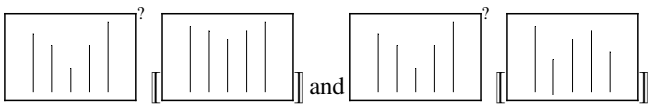
We now consider another usage, which is more general and more tricky: Consider the predicate

Definition["Tricky Order", any[standard, A],
is–ordered[standard, A] : ⟺
$\forall_{i=2,\ldots,|standard|-1} \; \forall_{j=1,\ldots,i-1}$ (standard$_i$ ≥ standard$_j$ ⟺ A$_i$ ≥ A$_j$)]

This predicate yields "True" iff the given list 'A' is ordered exactly in the way as specified by the ordering of the list 'standard'. Now, it may be interesting to introduce an extra logicographic symbol for each concrete value of the parameter 'standard'. In other words, the logicographic representation for the predicate 'is-ordered' would depend on the value of the first argument 'standard'. A possible choice would be, for example:

LogicographicNotation["4−ary limit", any[A, standard],
is–ordered[standard, A] (A " is ordered by" standard)
⇋ standard$^?$ ⟦A⟧];

Then



would yield "True" and "False" respectively.

3.2 Example: Graph Representation

One of natural applications of variable shape logicographic symbols is the graph theory. In the graph theory a graph can be defined as a pair of a set of vertices and a set of edges. For example, this is a graph which has 4 vertices with 3 edges

graph[{v1, v2, v3, v4}, {{v1, v2}, {v1, v3}, {v1, v4}}] .

Then the natural logicographic representation of the graph is '  '.

Obviously for humans it is much easier to understand formalized statements of the graph theory by this graphical representation than the formula representation. The following example, taken from a book [Gib1985], shows the computation sequence of chromatic polynomial of a graph.

By 'chromatic–polynomial[G,k]', represented logicographically by $[\![G]\!]_k$, we denote the number of ways of vertex-coloring the graph 'G' with 'k' colors such that verities of 'G' should not have the same color if they are adjacent by the edges of 'G'. For example,

$[\![$$]\!]_k = k\,(k-1)^3$

because the vertex in the center can be colored in k different ways. The remaining vertices can then be colored in (k–1) ways. Now here is a useful theorem to compute chromatic polynomials:

Theorem["7.10 in the book[Gib1985]", any[G, e],
edge[G, e] $\Longrightarrow [\![G]\!]_k = [\![G - e]\!]_k - [\![G \circ e]\!]_k$]

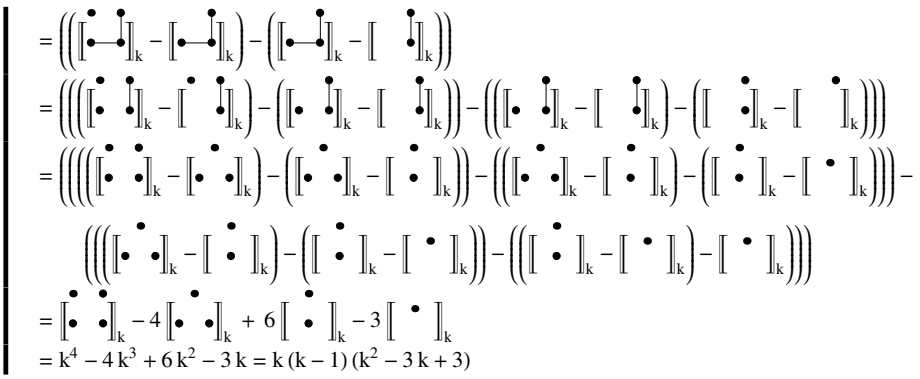
where 'edge[G,e]' means 'e' is an edge of 'G', 'G–e' is derived from 'G' by deleting the edge 'e', and 'G ∘ e' is obtained from G by contracting the edge 'e'. Intuitively contraction of an edge means making the vertices of the edge into one vertex.

We see the idea of the proof by an example. Let 'G' be '  ' and 'e' be '', by the theorem the following fact holds:

$[\![$  $]\!]_k = [\![$  $]\!]_k - [\![$  $]\!]_k$

Here $[\![$  $]\!]_k$ contains the all cases of $[\![$  $]\!]_k$ except the cases where the two upper vertices are the same color. And the number of the exceptional cases is exactly $[\![$  $]\!]_k$. Repeated application of this theorem will eventually reach to the combination of chromatic polynomials with no edges which can be easily computed.

$[\![$  $]\!]_k$
$= [\![$  $]\!]_k - [\![$$]\!]_k$



$$= k^4 - 4k^3 + 6k^2 - 3k = k(k-1)(k^2 - 3k + 3)$$

With this logicographic representation we will be able to show formal statements of the graph theory more intuitively, and also use the logicographic representation as formally manipulatable expressions, e.g. for computation.

3.3 Example: Geometrical Judgement

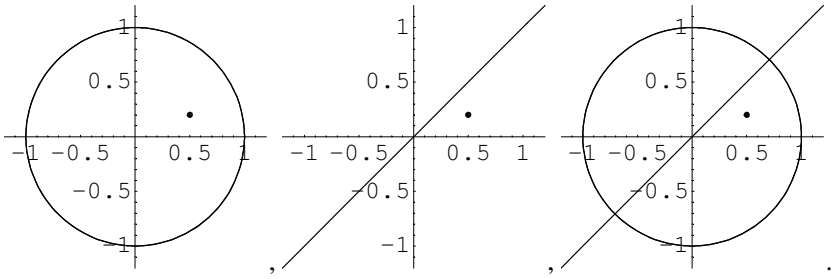
It is a natural idea to use a logicographic symbol for algebraic inequalities as geometrical representation. As an application, the judgement of the region inclusion can be converted into the judgement whether an inequality holds or not. For example, suppose we have the following definitions:

Definitions["in a region", any[G, e],
in–circle[x, y] : $\Longleftrightarrow x^2 + y^2 < 1$
under–line[x, y] : $\Longleftrightarrow y < x$]

Then the formulae

in–circle[0.5, 0.2], under–line[0.5, 0.2], (in–circle $\wedge$ under–line)[0.5, 0.2]

should be represented logicographically in the following ways respectively:



The important observation is '(in–circle$\wedge$under–line)[0.5,0.2]' which can be composed from the representations of 'in–circle' and 'under–line' by using the technique explained in Section 2.3. By seeing the picture, one can reason immediately the truth value of formulae without computing algebraic inequalities. (Of course when a point is very close to curves or lines, one can not judge by seeing.)

This kind of reasoning are related to the topic so called 'logical reasoning with diagrams'. One famous example is the Venn diagram by which one can reason some facts with the graphical operations. Shin[Shi1995] and Hammer[Ham1995] improved the Venn diagram and elaborated it in such ways that certain theorems are proved diagrammatically. They also gave soundness and completeness proofs. This is a counter argument against the prejudice that using diagrams is unreliable and that they should not be used in formal proofs. If it is well constructed, one does not lose formality at all. In more general settings the discussions of the properties which these reasoning systems should satisfy can be found in [BS1995, Shi2001].

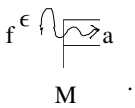
4. Designing Logicographic Symbols

Traditionally, one of the important rules for the invention of notation has always been that it should be easy and fast to write the notation on paper or blackboard. However this rule is not any more so important because inputting becomes faster and easier also for quite involved symbols if we use computers. With this new tool, the users of mathematical systems like Theorema now have a potentially unlimited collection of mathematical symbols at their disposition for expressing their ideas. The design of the logicographic symbol is completely free. However, there are some pedagogic, psychological and design principles which should be followed.

- **Should be as simple as possible and as complicated as necessary**

Most importantly, logicographic symbols should be designed as simple as possible and as complicated as necessary: If a symbol becomes too complicated, it is, again, hard to grasp the essential features of the notion quickly. If a symbol is too simple, it may omit essential features that distinguish the notion from other, similar, notions.

However, the exact design of a logicographic symbol may depend on the pedagogical situation of the addressee in a formal mathematical text: If the notion to be represented by a logicographic symbol is completely new to the addressee, it may be preferable to design a more involved logicographic symbol like the logicographic symbol for 'limit' in Section 2.1. If the notion becomes more and more familiar, an alternative, more concise version of the symbol may be introduced. In fact, it is perfectly possible to attach two or more logicographic symbols with one notion. For example, another possibility could be

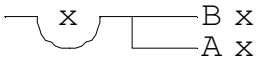


.

- **Should reflect the concept behind**

Secondly the logicographic symbols should reflect the concept behind. For example, we should not use 'ascendant triangle:  ' for the notion of the predicate checking whether a list is in descendant order or not.

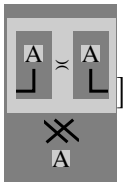
In 1879, Frege introduced 2-dimensional notation for logical formulae [Fre1967]. For example, the formula '$\forall_x (A[x] \Longrightarrow B[x])$' is described as follows:



Frege's notation, however, shows only the syntactical structure of formula, whereas logicographic symbols try to convey the intuitive semantics behind the logical constants. Obviously we could implement logicographic symbols for such notation. However the Frege's notation does not convey the intuitive semantics.

- **Should use vertical direction and colors**

Usually infix notations are located horizontally e.g 'lsp[A]≍rsp[A]'. On the contrary, in the example of 'merge sort' some logicographic symbols are located vertically, too. For example, an axiom 'splitting and concatenating gives a permuted version of the original' can be described with logicographic symbols as follows:

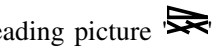
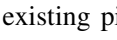
Axiom["csp" , any[A], 

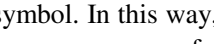
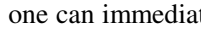
]

It uses vertical direction for '  ' which makes the structure clear with the help of colors and enhances the readability. If we do not use the vertical direction and colors, it would be represented as '  A' which does not make us understood immediately. Of course, the above logicographic representation consumes a lot of 2-dimentional space whereas traditional representation uses less 1-dimentional space. However, as mentioned, it is not a problem at all if we use computers, and always we can switch the logicographic symbol depending on the situation.

- **Composite symbols have the meaning of all continents**

The tool of composing representations helps for the notational conciseness. In some sense composed representation have the all properties of its constituents.

For example, in the 'merge sort' theory, the leading picture '  ' is combined by two existing pictures '  ' and '  ' which indicate that this expression is in some sense have both meanings of constituents as the definition of 'is sorted version'.

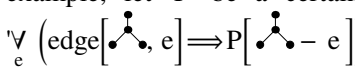
Another example in the proof of 'merge sort', the lemma "Bijective is Injective" is used twice. Note that one can see this fact immediately without referring the lemma because we constructed the logicographic symbols in such a way that the properties represented by a part of a logicographic symbol can be implied from the properties represented by the entire symbol. In this way, one can immediately notice the facts like '  ', '  ', '' etc. from '  '.

5. Conclusion

After reviewing the examples of fixed shape logicographic symbols, the idea of variable shape logicographic symbols which change their shapes depending on the arguments is proposed by some examples. There are many applications of variable logicographic symbols, e.g. graph theory, geometry, topology, category theory, term rewriting theory etc.

The problem of creating variable shape logicographic symbols is that the drawing and interpreting algorithm should be described on a case basis. Probably there should be a library which consists of some basic functions helping to draw pictures easily and interpreting meanings from drawing components like lines, circles etc.

There are many existing systems which can visualize mathematical objects. However these systems can not use these visualized representation for evaluation. With the tool of logicographic symbols one can freely interchange between visual thinking and logical thinking. This is a distinctive characteristic which can not be seen in other mathematical systems.

In mathematics there are some parts which visual description is suited for and other parts which logical description is suited for. As a tendency, visual description is suited for concrete examples and logical description is suited for describing general properties. For example, let 'P' be a certain property then it's difficult to describe the statement '  ' only by visual representation, because 'e' is not concrete yet.

We believe that the tool of logicographic symbols presented in this paper should be integrated into mathematical software systems in future.

References

[BDJ+2000] B. Buchberger, C. Dupre, T. Jebelean, F. Kriftner, K. Nakagawa, D. Vasaru, and W. Windsteiger. The THEOREMA Project: A Progress Report. In *CALCULEMUS 2000 (International Workshop on Systems for Integrated Computation and Deduction)*, St. Andrews, Scotland, August 2000.

[BJK+1997] B. Buchberger, T. Jebelean, F. Kriftner, M. Marin, E. Tomuta, and D. Vasaru. A Survey on the Theorema Project. In W. Kuechlin, editor, *ISSAC'97 (International Symposium on Symbolic and Algebraic Computation)*, pages 384–391, Maui, Hawaii, ACM Press, July 21-23 1997. Also available as RISC technical report 97-15.

[BS1995] J. Barwise and A. Shimojima, Surrogate Reasoning. Cognitive Studies: Bulletin of the Japanese Cognitive Science Society, Vol. 2, No. 4, pp.7-26. 1995.

[Buc2000] B. Buchberger. Logicographic Symbols: Some Examples of Their Use in Formal Proofs. RISC (Research Institute for Symbolic Computation), Johannes Kepler University, Linz, Austria, Feb. 2000. Manuscript.

[Buc2001] B. Buchberger. Logicographic Symbols: A New Feature in Theorema. In Y. Tazawa, editor, *Fourth International Mathematica Symposium (IMS 2001)*, Chiba, Tokyo Denki University. June 2001.

[Fre1967] G. Frege. Begriffsschrift, a Formula Language, Modeled upon that of Arithmetic, for Pure Thought. In *From Frege to Gödel: a Source Book in Mathematical Logic, 1879—1931*, pages 1—82. Harvard University Press, 1967. Original in German in 1879.

[Gib1985] A. Gibbons. Algorithmic Graph Theory. Cambridge University Press, 1985. p. 203.

[Ham1995] E. M. Hammer. *Logic and Visual Information*, CSLI Publications, Stanford, California, 1995.

[NB2001a] K. Nakagawa and B. Buchberger. Presenting Proofs Using Logicographic Symbols. In *Proceedings of the Workshop on Proof Transformation and Presentation*, page 11, Siena, Italy, 2001. http://www.scch.at/research/publications/522/index.html. PTP-01 in IJCAR-2001.

[NB2001b] K. Nakagawa and B. Buchberger. Two Tools for Mathematical Knowledge Management in Theorema. In B. Buchberger and O. Caprotti, editor, *First International Workshop on Mathematical Knowledge Management (MKM 2001)*, Schloss Hagenberg, Austria, RISC. September 2001. http://www.scch.at/research/publications/702/index.html.

[Nak2002] K. Nakagawa. Supporting User-Friendliness in the Mathematical Software System Theroema. Ph.D. thesis of Research Institute for Symbolic Computation, January 2002.

[Shi1995] S. Shin. *The Logical Status of Diagrams*, Cambridge University Press, Cambridge, England, 1995.

[Shi12001] A. Shimojima, A Logical Analysis of Graphical Consistency Proof. Abstracts of the International Conference of Model-Based Reasoning, p. 41, 2001.

Solving Linear Boundary Value Problems via Non–Commutative Gröbner Bases*

MARKUS ROSENKRANZ[1] AND HEINZ W. ENGL[2]

[1] *Research Institute for Symbolic Computation, Kepler University, A–4020 Linz, Austria*
[2] *Institute for Industrial Mathematics, Kepler University, A–4020 Linz, Austria*

* *This work was supported by the Austrian Science Foundation FWF under the SFB grants F1302 and F1308.*

Abstract

A new approach for symbolically solving linear boundary value problems is presented. Rather than using general–purpose tools for obtaining parametrized solutions of the underlying ODE and fitting them against the specified boundary conditions (which may be quite expensive), the problem is interpreted as an operator inversion problem in a suitable Banach space setting. Using the concept of the oblique Moore–Penrose inverse, it is possible to transform the inversion problem into a system of operator equations that can be attacked by virtue of non–commutative Gröbner bases. The resulting operator solution can be represented as an integral operator having the classical Green's function as its kernel. Although, at this stage of research, we cannot yet give an algorithmic formulation of the method and its domain of admissible inputs, we do believe that it has promising perspectives of automation and generalization; some of these perspectives are discussed.

KEYWORDS: Symbolic Methods for ODE, Linear BVP, Moore–Penrose Equations

1. Introduction

Sophus Lie said in 1894 what is nowadays folklore [13, p. 488]: "All branches of physics pos problems that end up in integrating differential equations," and similar things can be said about many other sciences. A great deal of these differential equations come in the form of *boundary value problems*, and it is this problem type that has inspired rich parts of functional analysis, as one can see nicely in the classic work of Hilbert–Courant [9].

It is therefore natural to ask about *symbolic methods* for boundary value problems (BVP). But quite in contrast to the rich arsenal of numerical algorithms for BVP, this corner of mathematics seems to be a bit neglected by the "symbolic world". Of course, there are some standard techniques available for various kinds of differential equations—ordinary and partial, linear and nonlinear [10][23][19]. At the first glance, one might think this is sufficient since one can always solve

corresponding differential equation and adapt the free coefficients of the generic solution to fit the boundary conditions. However, we are not only asking for the solution of one individual differential equation generated by fixing the inhomogeneity on its right–hand side; what we want is a generic expression that can be instantiated by all admissible right–hand sides for producing the corresponding solutions (see below). Besides this, the generic solution might have no closed form whereas its "adaption" to the given boundary conditions often does.

Therefore we propose a new approach that works on the BVP as a whole, representing both the differential equation and the boundary conditions by operators on suitable Banach spaces. Such a *functional–analytic setting* is of course very familiar in abstract convergence analysis of numerical BVP algorithms, but interestingly it turns out to be equally useful for searching symbolic solutions via non–commutative Gröbner bases. The idea is that both the differential and the boundary operator are built up from some "atomic" operators and can thus be seen as non–commutative polynomials with the atomic operators as its indeterminates. For obtaining suitable polynomial equations, we use the powerful concept of the oblique Moore–Penrose inverse [21].

In this paper, we consider only *ordinary* differential operators and *linear* BVP; see Section 4 for a discussion of possible extensions. Furthermore we will search for solutions over a *finite interval* $[a, b]$. Now let T be a linear differential operator of order n, so for $u \in C^n[a, b]$ we have

$$T\,u = c_0\,u^{(n)} + \ldots + c_{n-1}\,u' + c_n\,u\,,$$

where $c_0, \ldots, c_n$ are sufficiently smooth coefficient functions (for example, $c_j \in C^{n-j}[a, b]$ for each $j = 0, \ldots, n$) and c_0 does not vanish. We view T as a linear operator on the Banch space $(C[a, b], \|\cdot\|_\infty)$ with dense domain of definition $\mathcal{D}(T) = C^n[a, b]$. The boundary operators $B_1, \ldots, B_n$ are defined on the same domain; for each $i = 1, \ldots, n$ we have

$$\begin{aligned} B_i\,u = {} & p_{i,0}\,u^{(n)}(a) + \ldots + p_{i,n-1}\,u'(a) + p_{i,n}\,u(a) + \\ & q_{i,0}\,u^{(n)}(b) + \ldots + q_{i,n-1}\,u'(b) + q_{i,n}\,u(b)\,, \end{aligned}$$

where the coefficients $p_{i,j}, q_{i,j}$ are real numbers. Now the *boundary value problem induced by T and $B_1, \ldots, B_n$* is to find for each right–hand side $f \in C[a, b]$ a function $u \in C^n[a, b]$ such that:

$$\begin{aligned} & T\,u = f \\ & B_1\,u = \ldots = B_n\,u = 0 \end{aligned} \qquad (1)$$

This BVP is actually inhomogeneous in the differential equation and homogeneous in the boundary conditions (*semi–inhomogeneous problem*). But we can always decompose a fully inhomogeneous problem into such a semi–inhomogeneous one and a rather trivial BVP with homogeneous differential equation and inhomogeneous boundary conditions (*semi–homogeneous problem*); see [24, p. 43] for an explanation. Furthermore, we will assume throughout the paper that the boundar conditions are such that they determine a unique solution u of (1) for all $f \in C[a, b]$.

We are now searching for an operator G that takes the inhomogeneity f as input and produces the solution u of (1) as output. In fact, in those cases which we consider, it is well–known that the operator G can be written as an integral operator with the so–called *Green's function* g as its kernel [8, p. 296]:

$$G f(x) = \int_a^b g(x, \xi)\, f(\xi)\, d\xi \qquad (2)$$

The desired solution operator G is obviously a *right inverse* of the given differential operator: $T\,G(f) = f$ and hence $T\,G = 1$. (For the sake of simplicity, we will use the symbol 1 for denoting various identity functions and operators.) Of course, there are many right inverses for T, but the boundary conditions $B_1\,u = \ldots = B_n\,u = 0$ are supposed to single out the one we want. It should be noted that this viewpoint is different from the standard one, where the boundary conditions are used for specifying the domain of the differential operator; in this case, there is of course only one inverse.

So we want to find a right inverse that is normally not an inverse in the strict sense—this is where the concept of the *oblique Moore–Penrose inverse* enters the stage (see Section 2 for details): Given the operator T on the Banach space $C[a, b]$ together with arbitrary projectors P, Q onto its nullspace and range closure, the oblique Moore–Penrose inverse $T^{\dagger}_{P,Q}$ can be determined by the four well–known Moore–Penrose equations, which can be seen as four non–commutative polynomial equations in the indeterminates $T, T^{\dagger}, P, Q$. By choosing suitable projectors P, Q, it may be possible to enforce the boundary conditions, which has the consequence that $T^{\dagger} = G$. In general, the projectors will thus become polynomials in $B_1, \ldots, B_n$ and some extra operators describing their particular structure. In many cases, one will be able to express some or all of the boundary operators as well as the differential operator T in terms of these extra operators. So let $A_1, \ldots, A_m$ be those boundary and extra operators that are needed; we will collectively call them *auxiliary operators*. Substituting them in the Moore–Penrose equations, we will end up with an equation system

$$\forall_{i=1,\ldots,4}\ \mathcal{P}_i(G, A_1, \ldots, A_m) = 0\,, \qquad (3)$$

where $\mathcal{P}_1, \ldots, \mathcal{P}_4$ are some non–commutative polynomials in the indicated indeterminates.

Our goal is to obtain a partial triangularization this system, i.e. to find an equivalent system containing an equation of the form $G = \ldots$, where the right–hand side should not contain G. This means we want a term representation for the solution operator G: it should be described in terms of some elementary operators like integration and multiplication. For giving a complete specification, we must therefore decide which *elementary operators* $E_1, \ldots, E_k$ we want to allow in the solution term for G. Depending on this choice, the task of triangularizing the equation system may be easy, difficult or even impossible. This is one of the critical points in our approach that should become algorithmic in the future (see Section 4 for a brief discussion of this topic): We must either be creative in finding "good" elementary operators or we need powerful structure theorems for warranting the completeness of certain basis operators.

Assuming we have established a suitable collection of elementary operators $E_1, \ldots, E_k$, we must still specify how they are related with the auxiliary operators $A_1, \ldots, A_m$ occurring in the Moore–Penrose equations, i.e. we need some polynomial equations that describe their interaction. For example, if E_1 is integration and A_1 is differentiation, the obvious relation between them is the Fundamental Theorem of Calculus. This step is the second half of the "creative" phase just described; both steps should be taken together. Having found enough *interaction equations*

$$\forall_{i=1,\ldots,l}\ \mathcal{Q}_i(A_1, \ldots, A_m, E_1, \ldots, E_k) = 0\,, \qquad (4)$$

we can combine (3) and (4), looking at it as a well–known problem of computer algebra: Given the ideal J induced by the polynomials $\mathcal{P}_1, \ldots, \mathcal{P}_4, \mathcal{Q}_1, \ldots, \mathcal{Q}_l$, try to find a basis for J containing a polynomial with leading term G; see at the end of Section 2 for an example. Having such a basis, we can write the corresponding equation in the desired form $G = \mathcal{G}(A_1, \ldots, A_m, E_1, \ldots, E_k)$, where $\mathcal{G}$ is a polynomial in the indicated indeterminates. If we have chosen suitable operators $A_1, \ldots, A_m, E_1, \ldots, E_k$, we can interpret the solution operator G as the usual Green's operator and extract from it the Green's function g.

For finding the desired basis, we use the *method of Gröbner bases*, introduced by the second author in his PhD thesis [3]; see also the journal version [4] and a concise treatment in [6]. The advantage of Gröbner bases is that they do not only lead to the desired solution but they also reveal useful information about the ideal structure. In this paper, however, we will not address these issues. For a modern survey of the theory of Gröbner bases and their applications, see [7] and the remarks at the end of this section.

The idea of using the *Moore–Penrose inverse for solving linear BVP* is not new. One can find a standard treatment of this subject in [22] and [24]. But what is new, to our knowledge, is the observation that by means of non–commutative Gröbner bases one can actually fertilize the Moore–Penrose equations for obtaining symbolic solutions. There is an interesting paper [18] from the seventies that describes a different Moore–Penrose method for approaching linear BVP. It is based on the concept of adjoint operators and orthogonal projectors (as opposed to the oblique ones used in our method), but it does not make use of Gröbner bases. This approach seems to result into more complex computations than ours, but it would be an interesting research topic to combine the two methods.

Non–commutative Gröbner bases have been applied to differential operators for several decades, see for example the survey article [24] about Gröbner bases and partial differential equations. However, most of the theory in this field is concerned with studying the structure of solutions, without giving explicit methods for constructing them (the situation becomes even worse when it comes to BVP). Besides this, Gröbner bases have been used for *simplifying complicated operator expressions* as they typically arise in control theory. This approach is described in the papers [15][16][26] of the San Diego group, which also served as the starting point for our investigations. used the software package developed by their group for the Gröbner–basis computations necessary in our examples; see Section 2 for details.

The difference between the problem considered here and the subject of simplification addressed by their group is of a fundamental nature. Applications of Gröbner bases—both in the commutative and in the non–commutative cases—come in *three main categories* [6]:

- *Confluent Rewriting*: A Gröbner basis induces a rewrite system for reducing polynomials. Using a suitable term ordering, this will sometimes lead to a drastically simpler optical appearance, which is very important for control theorists [26]. However, the essential poir is that the reduced form is not only optically simpler but even canonical, due to the characteristic Church–Rosser property of Gröbner basis. This means that one can decide equality: Two polynomials are equal in the given ideal if and *only* if their reduced forms are identical.

- *Polynomial Equation Solving*: Using a term ordering of the lexicographic type, Gröbner bases enjoy the so–called elimination property. Basically this means that the equation

system will be triangularized as much as possible so that it is easy to solve the resulting system. The elimination property also holds in the non–commutative case; see [2].

- *Syzygies*: The information contained in the reductions that transform a given set of polynomi als into a Gröbner basis can be used to determine the complete solution module of a linear equation system over a polynomial ring.

Seen in this way, research in the San Diego group belongs to the first category whereas our research belongs to the second. It might be worthwhile to also carry out operator–theoretic investigations in fields pertaining to the third application category.

The rest of the paper is *structured as follows*: In Section 2 we take a well–known linear BVP as a simple but yet interesting example for walking through the whole procedure outlined above. In Section 3 we briefly present some more examples demonstrating different boundary conditions and slightly more complicated differential equations. In Section 4 we conclude with some reflections about the methodology and the potential of automation and generalization.

2. A Detailed Computation

The following problem seems to be one of the classical examples that are most often used for introducing the concepts of linear BVP [24, p. 42]. It can be interpreted as describing *one–dimen sional steady heat conduction in a homogeneous rod*. We will discuss this example in some detail for illustrating the solution strategy presented in the previous section.

Given: $f \in C[0, 1]$,

find: $u \in C^2[0, 1]$

such that

$$u'' = f \,,$$
$$u(0) = u(1) = 0 \,.$$

The general problem described above is now given the simple instantiation $[a, b] = [0, 1]$, $n = 2$, $T = D^2$, $B_1 = L$, $B_2 = R$. Here D^2 denotes the iterated differentiation operator on the Banach space $(C[0, 1], \|\cdot\|_\infty)$; it has the subset $C^2[0, 1]$ as its dense domain of definition. The left and right boundary operators L, R are defined in the obvious way: For each $u \in C[0, 1]$, we have $Lu = u(0)$ and $Ru = u(1)$. As described above, we interpret this as an *inversion problem* in the following sense: Find a right inverse G of the operator D^2 such that the boundary conditions are also fulfilled. We construct G as a Moore–Penrose inverse.

From the theory [12, p. 567] it is clear that we must fix *appropriate projectors* P and Q onto th nullspace and range–closure of D^2, respectively. The latter will always be the identity $1 : C[0, 1] \to C[0, 1]$ for the type of problems considered here; as a consequence, G is bounded and defined on all of $C[0, 1]$. The other projector P maps $C[0, 1]$ onto the *nullspace*

$$N = \mathcal{N}(D^2) = \{x \mapsto \alpha\, x + \beta \mid \alpha, \beta \in \mathbb{R}\},$$

so choosing P amounts to specifying for each $u \in C[0, 1]$ real numbers α, β such that $(P\,u)(x) = \alpha\, x + \beta$ for all $x \in [0, 1]$. We use this freedom to ensure the boundary conditions, which leads to

$$P = (1 - X)\, L + X\, R\ ,$$

where X is defined as the operator induced by multiplication with the independent variable.

Substituting this together with Q into the general equations in [12, p. 567], we arrive at th following *concrete Moore–Penrose equations*:

$$\begin{aligned} &D^2\, G\, D^2 = D^2 \\ &G\, D^2\, G = G \\ &G\, D^2 = 1 - (1 - X)\, L - X\, R \\ &D^2\, G = 1 \end{aligned} \tag{5}$$

We can see that they form indeed a *system of polynomial equations*, having the desired Green's operator G and the auxiliary operators (named $A_1, \ldots, A_m$ in the introduction) D, X, L, R as indeterminates. The only thing missing now are the elementary operators (named $E_1, \ldots, E_k$ in the introduction) that we want to allow in the solution term, together with suitable relations describing their interaction with the auxiliary operators.

Now we come to the "creative" step of our approach (see Section 4 for a brief discussion on the potential of automation). It is clear that the operators D, X, L, R will not be sufficient for expressing the solution term for the Green's operator G. Since we would like to have an integral representation for G, having the corresponding Green's function g as its kernel, we must obviously take the *antiderivative operator* A as one elementary operator. It is defined in the obvious way as

$$(A\,u)\,x = \int_0^x u(\xi)\, d\xi$$

for all $u \in C[0, 1]$ and $x \in [0, 1]$. What other elementary operators might be needed? In view of the duality in the boundary operators L, R, we may have the idea of adding the operator B *adjoint to the antiderivative operator* A. Whereas the operator A integrates *from* the left boundary, the operator B integrates *to* the right boundary, so it is defined as

$$(B\,u)\,x = \int_x^1 u(\xi)\, d\xi$$

again for all $u \in C[0, 1]$ and $x \in [0, 1]$. Having A and B as elementary operators along with the auxiliary operators D, X, L, R, it turns out that we can express the solution G in the desired way. The following *interaction equations* are sufficient for describing their relations:

$$\begin{aligned} &D\, X = X\, D + 1 \\ &D\, A = 1 \\ &A\, D = 1 - L \\ &D\, B = -1 \end{aligned} \tag{6}$$

$$B\,D = R - 1$$
$$R\,X = R$$
$$L\,X = 0$$

At this point, we have assembled the complete polynomial equation system, consisting of the polynomials $\mathcal{P}_1, \ldots, \mathcal{P}_4$ in the concrete Moore–Penrose equations (5) and the polynomials $\mathcal{Q}_1, \ldots, \mathcal{Q}_7$ of the interaction equations (6). Our goal is to solve this system for G, i.e. we want to find the elimination ideal with respect to G. For this we will use the following *multigraded lexicographic term ordering*:

$$D < R < L < X < A < B \ll G$$

For computing the desired elimination ideal, we use the system *NCAlgebra* [14], a Mathematic package for doing non–commutative computer algebra, written by J. William Helton (Mathematics Department of the University of California, San Diego, California) and Robert L. Miller (General Atomic Corporation, La Jolla, California). It includes support for non–commutative Gröbner bases, also described in the papers [15][16][26]. Typically, we must content ourselves with a partial basis, this is sufficient for us as long as G is isolated. For the current problem *NCAlgebra* returns the following answer:

```
final = NCMakeGB[initial, 2] // ColumnForm
−1 + D ** A
−1 + L + A ** D
1 + D ** B
1 − R + B ** D
1 − D ** X + X ** D
−R + R ** X
L ** X
L ** A
−A − B + R ** A
−A − X + D ** X ** A
…
G + A ** X + X ** B − X ** A ** X − X ** B ** X
−R + L ** R
−R + D ** X ** R
−A − B + B ** A + B ** X + X ** A
…
```

The system has produced 42 polynomials, most of which are left out above as they are not interesting for our present purposes; e.g. some of them express integration rules for polynomials such as $2\,x^2 + 3\,x$. The only important thing is that there is only *one polynomial involving the solution operator* G, and in this polynomial, G does indeed occur isolated. Writing the result in the usual format, we arrive at:

$$G = X\,A\,X - A\,X + X\,B\,X - X\,B$$

It is straightforward to rewrite this polynomial into the traditional formulation of the corresponding *Green's function*:

$$g(x, \xi) = \begin{cases} (x-1)\,\xi & \text{if} \quad 0 \le \xi \le x \le 1 \\ x(\xi - 1) & \text{if} \quad 0 \le x \le \xi \le 1 \end{cases}$$

3. Other Examples

Passing on to other examples, let us first remark that we can use *various other types of boundary conditions* without making essential changes in the computation just presented. For example, using the mixed conditions $u'(0) = u(1) = 0$ will lead to the nullspace projector

$$P = X\,L\,D + R + L\,D\,,$$

whereas the conditions $u(0) = u'(0) = 0$ will lead to the nullspace projector

$$P = X\,L\,D + L\,.$$

Everything else remains the same, and the computation results in the correct Green's functions for these cases. (Specifying the boundary conditions $u'(0) = u'(1) = 0$, however, would not allow a unique solution for all right–hand sides $f \in C[0, 1]$. In fact, one can apply the well–known *Fredholm alternative* for characterizing solvability in such cases. If we tried to apply our method to such a case, we would end up with redundant parameters. New ideas are necessary for dealing with these cases, but we will not address them here.)

For a slightly more complicated problem, we take Example 2 in Kralle's book [17, p. 109]. Th differential operator of this BVP has *damped oscillations* as its eigenfunctions [17, p. 107]. Stated i our terminology, the problem reads as follows:

$$-\frac{(e^{2x}\,u(x)')' + e^{2x}\,u(x)}{e^{2x}} = f(x)$$
$$u(0) = u(\pi) = 0$$

Here x is assumed to range over the interval $[0, \pi]$. The notation $\boldsymbol{T}'$ is an abbreviation for $\frac{d}{dx}\,\boldsymbol{T}$, where the differential quantifier $\frac{d}{dx}$ operates on the term $\boldsymbol{T}$. For obtaining an operator equation, we introduce some *auxiliary operators*. For all $u \in C[0, \pi]$, $v \in C^1[0, \pi]$ and $x \in [0, \pi]$, we define:

$$(D\,v)(x) = v(x)'$$
$$(E\,u)(x) = e^x\,u(x)$$
$$(F\,u)(x) = e^{-x}\,u(x)$$
$$(L\,u)(x) = u(0)$$
$$(R\,u)(x) = u(\pi)$$

Using these operators, the given BVP can be stated in the following *operator–theoretic form*:

$$-(F^2\,D\,E^2\,D+1)\,u=f$$
$$L\,u=R\,u=0$$

Going through the procedure explained above, one finds for the nullspace projector

$$P=\frac{e^{\pi}}{\pi}\,X\,F\,R-\frac{1}{\pi}\,X\,F\,L+F\,L\,,$$

It turns out that one does not need other operators except A and B as in the previous examples, but one must add some obvious interaction equations for the new operators E and F. Carrying out the computation in *NCAlgebra*, one obtains the following result (after applying some tedious tricks for representing the "commuting variables" e, π):

$$G=F\,A\,X\,E+X\,F\,B\,E-\frac{1}{\pi}\,X\,F\,A\,X\,E-\frac{1}{\pi}\,X\,F\,B\,X\,E=$$
$$\frac{1}{\pi}\,(\pi-X)\,F\,A\,X\,E+\frac{1}{\pi}\,X\,F\,B\,(\pi-X)\,E$$

This time, the partial basis contains 164 polynomials, but there is still only one among which involves G, namely exactly the one corresponding to the solved equation above. Going through the usual translation procedure, one can write G as an integral operator with the following *Green's function* also given in [17, p. 110]:

$$g(x,\xi)=\begin{cases}\frac{1}{\pi}\,(\pi-x)\,\xi\,e^{\xi-x} & \text{if}\quad 0\le\xi\le x\le\pi\\ \frac{1}{\pi}\,(\pi-\xi)\,x\,e^{\xi-x} & \text{if}\quad 0\le x\le\xi\le\pi\end{cases}$$

The method presented here is not restricted to the classical setting of second–order Sturm–Liouville theory. For seeing this, we take a practically relevant fourth–order problem [17, p. 49], whic describes the *transverse deflection* $u\in C^2[0,1]$ of a homogeneous beam with distributed transversal load $f\in C[0,1]$, simply supported at both ends:

$$u^{(4)}=f$$
$$u(0)=u(1)=u''(0)=u''(1)=0$$

Its *operator–theoretic formulation* is as follows:

$$D^4\,u=f$$
$$L\,u=R\,u=L\,D^2\,u=R\,D^2\,u=0$$

Comparing this BVP to the simple heat–conduction problem considered in the beginning, we observe a strong *similarity*. In fact, the only difference is the order of the differential operator and the additional boundary conditions for u'', so we expect that we can use the same auxiliary and elementary functions.

This expectation is indeed fulfilled. Going through the same procedure as in the heat–conduction example, the boundary conditions lead to the following *nullspace projector*:

$$P = \frac{1}{6} X^3 (R D^2 - L D^2) + \frac{1}{2} X^2 L D^2 + \frac{1}{6} X (6 R - 6 L - 2 L D^2 - R D^2) + L$$

Using this operator and the interaction equations from the heat–conduction problem, we obtain a polynomial system that can be solved for G. The partial basis returned by *NCAlgebra* consists of 67 polynomials, and exactly one polynomial among them contains the indeterminate G for the *Green's operator*. Written as an equation, this polynomial is:

$$G = \frac{1}{3} X A X - \frac{1}{6} A X^3 - \frac{1}{2} X^2 A X + \frac{1}{6} X A X^3 + \\ \frac{1}{6} X^3 A X + \frac{1}{3} X B X - \frac{1}{2} X B X^2 - \frac{1}{6} X^3 B + \frac{1}{6} X B X^3 + \frac{1}{6} X^3 B X$$

As usual, we can immediately translate this expression to the more traditional formulation in terms of a *Green's function* g defined thus:

$$g(x, \xi) = \begin{cases} \frac{1}{3} x \xi - \frac{1}{6} \xi^3 - \frac{1}{2} x^2 \xi + \frac{1}{6} x \xi^3 + \frac{1}{6} x^3 \xi & \text{if} \quad 0 \le \xi \le x \le 1 \\ \frac{1}{3} x \xi - \frac{1}{2} x \xi^2 - \frac{1}{6} x^3 + \frac{1}{6} x \xi^3 + \frac{1}{6} x^3 \xi & \text{if} \quad 0 \le x \le \xi \le \pi \end{cases}$$

This is in full accordance with [17, p. 71], where the result was obtained by means of the causa fundamental solution.

4. Conclusion

We have presented a new *method for solving linear boundary value problems by symbolic techniques*. It proceeds by transforming the given differential equation and its boundary conditions into a system of polynomial equations that can be solved for the desired Green's operator via non–commutative Gröbner bases. Of course, one must specify those operators and properties that should be used for representing the solution term; using the traditional framework of integral operators, one obtains a solution in terms of the usual Green's function. For several examples, we have exhibited suitable interaction equations that lead to a Green's formulation of the solution. (Incidentally, we have also found other representations of the solution, typically involving multiple integrations. Though outside the framework of the traditional Green's functions, these representations may be of numerical interest due to their smoothness properties.)

Let us now briefly analyze the *current status of algorithmization* in this method. In a typical application, it will proceed through the following steps:

- *Derivation of the concrete Moore–Penrose equations*: The major task in this step is to determine the nullspace projector P since we have seen that the range projector Q is always the identity. Substituting P, Q and the given differential operator T in the generic Moore–Penrose equations and renaming the Moore–Penrose inverse $T^\dagger$ into G, we obtain the concrete Moore–Penrose equations. The polynomial for P will contain various auxiliary operators $A_1, \ldots, A_m$, usually made up from parts of the differential and boundary operators.

- *Compilation of the interaction equations*: After selecting suitable elementary operators $E_1, \ldots, E_k$, we have to find sufficiently many equations describing the relations between the auxiliary operators $A_1, \ldots, A_m$ and the elementary operators $E_1, \ldots, E_k$.

- *Computation of a partial Gröbner basis*: The concrete Moore–Penrose equations are combined with the interaction equations and supplied to a non–commutative Gröbner basis system, using a term ordering that isolates the Green's operator G.

- *Extraction of the Green's function*: The Green's operator G obtained in the previous step is transformed into the corresponding Green's function g.

For the first step and the last two steps, our method can be viewed as an algorithm (relative to the solvability of the homogeneous differential equation). For the second step, some *ad–hoc inventions* are still necessary for each problem instance at hand. In particular, one has to provide suitable interaction equations for specifying the solution structure. Some experience in handling BVP should be sufficient for finding these equalities.

Note that, after having found some basic interaction equations, the question of *how and in which order* these equations should be applied is exhaustively answered by the method of Gröbner bases, due to their Church–Rosser property. In a manual calculation, one has to play around with many possible ways of combining equations, which may or may not lead to success. In this sense, an essential portion of the usual "tricks" occurring in manual calculations is covered by our method; the remaining tricks are assoiated with the interaction equations.

We believe that our method can cover various interesting classes of BVP, which we plan to explore in forthcoming papers (including some of the generalizations discussed below). In an ideal situation—presumably hard to achieve—one might approach a systematic search of elementary operators and interaction equations in a manner similar to the *structure theorems* of Liouville theory, which are used for indefinite integration and ordinary differential equations [10, p. 186]. Fixing certain algebraic input domains (e.g. the elementary transcendental functions) for the coefficients of the differential and boundary operators, one might be able to isolate a suitable "Green's domain" $\mathfrak{G}$ such that the Green's function g will always be in $\mathfrak{G}$. We think that such an expectation is realistic because it is well–known that one can express g in terms of solutions of certain initial value problems. (Note also that we do not claim that the solution of the BVP itself, namely $G f$ for a given right–hand side f, should have any algebraically simple form.)

Having found a Green's domain $\mathfrak{G}$, it is probably not difficult to isolate appropriate *elementary operators* along with their *interaction equations*. Some of these operators might be multiplication operators induced by functions from the input domains and $\mathfrak{G}$, similar to F and G in Section 3. It should also be observed that most of the interaction equations considered in this paper would come out quite naturally when the elementary operators are introduced in systematic exploration cycles as described in [5].

Finally, let us propose further lines of future research. The problems considered in this paper have some obvious generalizations. Increasing the number of independent variables leads us to BVP for *partial differential equations* like the well–known Dirichlet problem for $\operatorname{div}(a \operatorname{grad} u) = f$. These equations will typically involve differential operators div, grad, rot, … from vector analysis: We

can either regard them as operators in their own right or assemble them from the partial differential operators ∂_x, ∂_y, ∂_z. The general methodology presented in this paper should be applicable in both cases; identities like the Divergence Theorem of Gauss will take over the role of the Fundamental Theorem of Calculus.

Passing over to *non–linear problems* is a much bigger challenge. In this case, compound operators may not be expressible as polynomials anymore. For example, take the non–linear operator $Q(u)(x) = u'(x)\,u(x)$. One would like to write this operator in terms of an elementary multiplication operator $M(u, v)(x) = u(x)\,v(x)$ as $Q(u) = M(D(u), u)$. Then we would have the product rule as an interaction equation $D \leftrightarrow M$, namely $D\,M(u, v) = M(D(u), v) + M(u, D(v))$. But this is not a purely operator–theoretic description anymore since we cannot get rid of u and v. This means general rewriting is necessary now: we need substitution in addition to replacement (reduction of polynomials is replacement on equivalence classes). Maybe this could be handled by a combination of Gröbner bases and the Knuth–Bendix algorithm. Actually this is a rather subtle topic, but there are promising results recently [1][20].

Orthogonal to these generalizations, one could also investigate *weak solutions*. In this paper, we have only considered classical solutions, but the results also make sense in a more general Sobolev setting. On the one hand, this simply changes domain and codomain of some operators; this does not harm the polynomial formulation since it abstracts from all topological notions. On the other hand, the solution concept itself must be modified by introducing suitable testing functions v and partial integrations. Logically this means that we have a universal quantifier over v on top of the equations, so we cannot take v as an indeterminate. Again, new ideas are necessary.

Apart from these generalizations, there is another issue that may be worth investigating. We have already observed after Equation (5) that the concept of polynomials is not fully adequate for capturing operator composition since it does not restrict the admissible combinations. This becomes even more apparent when we introduce operators like div and grad. In this case, we would like to distinguish vectors from scalars. For example, the composition div grad is admissible whereas div div does not make sense. But the question of domain adequacy is not of a purely aesthetic nature: It would prevent a great deal of unnecessary S–polynomials during the search for a Gröbner basis. We would need a notion of *restricted polynomials in* $X_1, \ldots, X_n$ such that each indeterminate X_i has an associated domain, $\mathrm{dom}(X_i)$, and codomain, $\mathrm{cod}(X_i)$, where we can build up monomials $X_i\,X_j\,X_k \cdots$ only if $\mathrm{dom}(X_i) = \mathrm{cod}(X_j)$ and $\mathrm{dom}(X_j) = \mathrm{cod}(X_k)$, etc. Since the structure of restricted polynomials is, by its very intention, not closed under multiplication, it figures as an algebraically rather unwieldy concept. It would be interesting to develop some alternative that combines practical needs and algebraic elegance.

References

[1] L. Bachmair, H. Ganzinger. Buchberger's algorithm: A constraint–based completion procedure. In: J.–P. Jouannaud, editor, First International Conference on Constraints in Computational Logics. Volume 845 of Lecture Notes in Computer Science, Springer Verlag, München, 1994, pp. 285–301.

[2] M. A. Borges, Mijail Borges. Gröbner Bases Property on Elimination Ideal in the Noncommutative Case. In [7], pp. 323–337.

[3] B. Buchberger. An Algorithm for Finding a Basis for the Residual Class Ring of Zero–Dimensional Polynomial Ideal (German). PhD Thesis, University of Innsbruck, Institute for Mathematics, 1965.

[4] B. Buchberger. An Algorithmic Criterion for the Solvability of Algebraic Systems of Equations (German). *Aequationes Mathematicae* 4, 1970, pp. 374–383.

[5] B. Buchberger. Theory Exploration Versus Theorem Proving. Invited talk at the Calculemus '99 Conference, Trento, Italy, July 1999. Available as RISC technical report at ftp://ftp.risc.uni–linz.ac.at/pub/techreports/1999/99–46.tar.gz .

[6] B. Buchberger. Introduction to Gröbner Bases. In [7], pp. 3–31.

[7] B. Buchberger, Franz Winkler (eds.). Gröbner Bases and Applications. London Mathematical Society, Lecture Note Series 251, Cambridge University Press 1998.

[8] E. A. Coddington, N. Levinson. Theory of Ordinary Differential Equations. McGraw–Hill Book Company, New York, 1955.

[9] R. Courant, D. Hilbert. Die Methoden der mathematischen Physik, Volumes 1 / 2. Springer Verlag, 4th edition, 1993.

[10] J.H. Davenport, Y. Siret, E. Tournier. Computer Algebra. Academic Press, London 1988.

[11] H.W. Engl, M. Hanke, and A. Neubauer, Regularization of Inverse Problems, Kluwer, Dordrecht, 1996.

[12] H. W. Engl, M. Z. Nashed. New Extremal Characterizations of Generalized Inverses of Linear Operators *Journal of Mathematical Analysis and Applications* **82**, 1981, pp. 566–586.

[13] G. Grosche, V. Ziegler, D. Ziegler. Teubner–Taschenbuch der Mathematik I. Teubner, Stuttgart, 1996.

[14] J. W. Helton, Robert L. Miller. The system NCAlgebra. Homepage at http://math.ucsd.edu/~ncalg, manual at http://math.ucsd.edu/~ncalg/NCBIGDOC/NCBIGDOC.html.

[15] J. W. Helton, Mark Stankus, John Wavrik. Computer Simplification of Engineering Systems Formulas IEEE Trans. Autom. Control 43, No. 3, 302-314 (1998).

[16] J. W. Helton, John Wavrik. Rules for Computer Simplification of the Formulas in Operator Model Theor and Linear Systems. *Operator Theory: Advances and Applications* **73** , 1994, 325—354.

[17] A. M. Krall. Applied Analysis. D. Reidel Publishing Company, Dordrecht, 1986.

[18] W. Loud. Some Examples of Generalized Green's Functions and Generalized Green's Matrices. *SIAM Review*, **12**(2), 1970.

[19] Y.–K. Man. Computing Closed Form Solutions of First Order ODEs Using the Prelle–Singer Procedure *Journ. Symb. Comp.* **16**, 1983, pp. 423–443.

[20] C. Marché. Normalized Rewriting: An Alternative to Rewriting Modulo a Set of Equations. *Journ. Symb. Comp.* **11**, 1996, pp. 1–36.

[21] M. Z. Nashed. Generalized Inverses and Applications, Proceedings of an Advanced Seminar Sponsored b the Mathematics Research Center, Universiy of Wisconsin–Madison, October 1973. Academic Press, New York 1976.

[22] M. Z. Nashed. Aspects of Generalized Inverses in Analysis and Regularization. In [21], pp. 193–244.

[23] M. J. Prelle, M. F. Singer. Elementary First Integrals of Differential Equations. *Trans. AMS*, **279**(1), 1983 pp. 215–229.

[24] D. C. Struppa. Gröbner Bases in Partial Differential Equations. In [7], pp. 235–245.

[25] I. Stakgold. Green's Functions and Boundary Value Problems. John Wiley & Sons, New York, 1979.

[26] J. Wavrik. Rewrite Rules and Simplification of Matrix Expressions. *Computer Science Journal of Mold ova*, **4** (2/11), 1996.

A Divide-and-Conquer Method for Integer-to-Rational Conversion *

Tateaki Sasaki[1], Yoshinori Takahashi[2] and Takuya Sugimoto[3]

[1]*Institute of Mathematics, University of Tsukuba, Tsukuba, Japan*
[2]*Master's Program in Education, University of Tsukuba*
[3]*Master's Program in Science and Engineering, University of Tsukuba*

Abstract

The integer-to-rational conversion is to find integers N and D for given relatively prime integers $M > 0$ and S, such that $DS \equiv N \pmod{M}$, $0 < D < \sqrt{M/2}$ and $|N| < \sqrt{M/2}$. The conversion plays an essential role in the modular computation of Gröbner bases over the rationals. This paper describes a divide-and-conquer method for the integer-to-rational conversion and compares it experimentally with Wang's algorithm and its enhanced version using Lehmer's technique. The experiment shows the superiority of the divide-and-conquer method clearly.

KEYWORDS: divide-and-conquer method, integer-to-rational conversion, modular Gröbner basis computation, rational number reconstruction

1. Introduction

Many algebraic algorithms performing successive arithmetic operations on polynomials with rational coefficients, such as algorithms for Gröbner basis computation, often cause extreme intermediate coefficients growth. In such cases, modular techniques are very useful. For example, Winkler [Win88] and Sasaki and Takeshima [ST89] proposed modular algorithms to compute Gröbner bases over the rationals; the former utilizes the Hensel lemma and the latter uses the Chinese remainder theorem. The latter algorithm, for example, proceeds as follows: 1) choose prime numbers $p_1, p_2, \ldots, p_n, p_{n+1}$ of word-size, 2) for $i = 1, 2, \ldots, n+1$, compute Gröbner basis modulo p_i, 3) construct a Gröbner basis modulo $M = p_1 p_2 \cdots p_{n+1}$ by the Chinese remainder theorem, 4) convert the

*Work supported in part by Japanese Ministry of Education, Science and Culture under Grants 12480065.

integer coefficients to rationals modulo M, and 5) check finally that the resulting Gröbner basis is a required one. Here, conversion of an integer to a rational modulo M is called *integer-to-rational conversion*.

Wang [Wan81] introduced the concept of integer-to-rational conversion (he called the conversion *rational number reconstruction*), and proposed an algorithm for it; see also [WGD82] and [SS92]. Wang's algorithm is the extended Euclidean algorithm with a special stopping condition. The algorithm is very simple and useful for integers of small and medium size, however, it is inefficient for integers of large size. On the other hand, integers of 1000 decimal digits or more appear frequently in actual applications, then the integer-to-rational conversion occupies a considerable part of the total computation time.

There are many studies on the integer GCD; see [Knu69] for old algorithms and [Sor94] for rather new algorithms. In particular, Schönhage [Sch71] proposed a divide-and-conquer method, and the method was applied to the univariate polynomial GCD by Moenck [Moe73]. (Schönhage's idea is easiest to understand in the univariate polynomial case, for which see [GG99], Sec. 11.1.) However, there are few studies on the integer-to-rational conversion, although Schönhage's idea is directly applicable to the integer-to-rational conversion.

Very recently, the present authors [STS01] and Pan and Wang [PW02] studied divide-and-conquering of the integer-to-rational conversion. We must analyze the mess of carries and the effect of truncation of integers, which are common to the GCD computation, and consider how to stop the remainder sequence computation at the bottom level of divide-and-conquering. In [PW02], a detailed analysis to solve these problems is given, which is different from ours, but it lacks a consideration on the "propagation of the errors" (see Subsection 4.4 for details). This paper is a complete version of [STS01].

After explaining the integer-to-rational conversion and Wang's algorithm in Section 2, we describe the idea of our divide-and-conquer method in Section 3. The details and verification of the method are given in Section 4. We have implemented our method in Lisp and compared it with Wang's algorithm and its enhanced version using Lehmer's technique; for Lehmer's technique, see [Knu69]. Although our implementation is a crude one, the comparison shows the superiority of the divide-and-conquer method clearly.

2. Integer-to-rational conversion

Let M and S be relatively prime integers, where $M > 0$. If there exist integers D and N satisfying

$$DS \equiv N \pmod{M}, \qquad 0 < D < \sqrt{M/2}, \quad 0 < |N| < \sqrt{M/2}, \tag{2.1}$$

then we say that the integer S is converted to the rational N/D modulo M. Note that N and D may not exist, in particular, for small M. It is well-known that integers D and N satisfying (2.1) and $\gcd(D, N) = 1$ are unique, so long as

they exist. If $N/D \equiv S \pmod{M}$ for $S > 0$ then $-N/D \equiv -S \pmod{M}$, and if $S \geq M/2$ then $N/D \equiv (S-M) \equiv -(M-S) \pmod{M}$. Therefore, without loss of generality, we assume in this paper that

$$M/2 > S > 0. \tag{2.2}$$

Remark 1: If M and S have a common divisor $g = \gcd(M, S) > 1$, then we search for integers D and N such that $S \equiv gN/D \pmod{M}$, or

$$D(S/g) \equiv N \pmod{M/g}, \quad 0 < D < \sqrt{M/2g}, \quad 0 < |N| < \sqrt{M/2g}. \tag{2.3}$$

Remark 2: In many practical cases, we can compute $\gcd(M, S)$ easily. In modular methods, we usually employ either the Hensel lemma or the Chinese remainder theorem (with Newton's interpolation method) to construct the resulting expression modulo M. In the Hensel construction, $M = p^{n+1}$ and $S = s_n p^n + \cdots + s_1 p + s_0$, where p is a prime and $p > s_i \geq 0$ $(i = 0, 1, \ldots, n)$. Hence, we can find $\gcd(M, S)$ at once. In the Chinese remainder construction, $M = p_{n+1} p_n \cdots p_1$ and $S = s_n p_n \cdots p_1 + \cdots + s_1 p_1 + s_0$, where $p_{n+1}, p_n, \ldots, p_1$ are usually primes and $p_{i+1} > s_i \geq 0$ $(i = 0, 1, \ldots, n)$. Therefore, we can easily calculate $\gcd(M, S)$ by dividing S by $p_1, p_2, p_3, \cdots$, successively. □

Wang's algorithm for the integer-to-rational conversion is as follows. Putting $S_0 = M$ and $S_1 = S$, we compute a remainder sequence $(S_0, S_1, S_2, \cdots, S_\kappa)$ and cofactor sequences $(C_0, C_1, C_2, \cdots, C_\kappa)$ and $(D_0, D_1, D_2, \cdots, D_\kappa)$ iteratively (the extended Euclidean algorithm):

$$\begin{cases} Q_i := \mathrm{quo}(S_{i-1}, S_i), & i = 1 \Rightarrow 2 \Rightarrow \cdots \Rightarrow \kappa - 1, \\ S_{i+1} := S_{i-1} - Q_i S_i & (\text{i.e., } S_{i+1} = \mathrm{rem}(S_{i-1}, S_i)), \\ C_{i+1} := C_{i-1} - Q_i C_i & \text{with } C_0 := 1 \text{ and } C_1 := 0, \\ D_{i+1} := D_{i-1} - Q_i D_i & \text{with } D_0 := 0 \text{ and } D_1 := 1. \end{cases} \tag{2.4}$$

Here, quo and rem denote quotient and remainder, respectively. Then, we have

$$C_i M + D_i S = S_i \quad (i = 0, 1, 2, \ldots, \kappa). \tag{2.5}$$

(Actually, the sequence $(C_0, C_1, C_2, \cdots, C_\kappa)$ is unnecessary). Note that each remainder S_i is positive due to (2.2). The sequence $(S_1, S_2, \cdots, S_\kappa)$ is descending, while the sequence $(|D_1|, |D_2|, \cdots, |D_\kappa|)$ is ascending. Hence, if we obtain the case that $|D_\kappa| < \sqrt{M/2}$ and $|S_\kappa| < \sqrt{M/2}$ then $D = |D_\kappa|$ and $N = \mathrm{sign}(D_\kappa)\, S_\kappa$, otherwise there exist no D and N satisfying (2.1). We list some properties of S_i, C_i and D_i for $S > 0$ and $i \geq 2$; see [GG99], Sec. 3.2.

$$S_i > 0, \quad \mathrm{sign}(C_i) = (-1)^i, \quad \mathrm{sign}(D_i) = (-1)^{i+1}, \tag{2.6}$$

$$|C_i| < |D_i|, \quad |C_i| < |C_{i+1}|, \quad |D_i| < |D_{i+1}|, \tag{2.7}$$

$$\text{with exceptions } \; C_2 \leq |D_2|, \;\; C_2 \leq |C_3|,$$

$$\begin{cases} S_0 = |D_i| S_{i-1} + |D_{i-1}| S_i & \Longrightarrow \quad |D_i| < S_0 / S_{i-1}, \\ S_1 = |C_i| S_{i-1} + |C_{i-1}| S_i & \Longrightarrow \quad |C_i| < S_1 / S_{i-1}. \end{cases} \tag{2.8}$$

3. A divide-and-conquer method

In this section, we describe an idea of divide-and-conquer method; details and verification of the method will be given in the next section.

First of all, we specify the representation of "big-integers". Let B be a word-size integer such that $B = 2^{32}$ or $B = 10^{16}$. We assume that S (and M, too) is represented as

$$\begin{aligned} &S = s_n B^n + s_{n-1} B^{n-1} + \cdots + s_1 B + s_0, \\ &s_n \neq 0, \quad B > s_i \geq 0 \ \ (i = n, n-1, \ldots, 0). \end{aligned} \tag{3.1}$$

We call S an n digit integer, and define $\mathrm{digit}(S)$ to be $\mathrm{digit}(S) = n$. We also call s_m the mth digit of S. Below, we put $\mathrm{digit}(M) = n$.

Our divide-and-conquer method is based on the following facts.

Fact 1: In the integer-to-rational conversion, we obtain D and N of about $n/2$ digits by eliminating about $n/2$ higher digits of M and S.

Fact 2: The cofactor sequences $(C_2, C_3, \cdots, C_\kappa)$ and $(D_2, D_3, \cdots, D_\kappa)$ can be computed by only the quotient sequence $(Q_1, Q_2, \cdots, Q_{\kappa-1})$, and the cofactor sequences allow us to compute remainder sequence $(S_2, S_3, \cdots, S_\kappa)$ by formula (2.5).

Fact 3: The quotient $Q_i = \mathrm{quo}(S_{i-1}, S_i)$ can be computed only by leading $2l+2$ digits of S_{i-1} and leading $l+1$ digits of S_i, where $l = \mathrm{digit}(S_{i-1}) - \mathrm{digit}(S_i)$.

Fact 4: Let $S_i > S_j > S_k > 0$. If we have two tuples of cofactors such that

$$\begin{aligned} (c'_u, d'_u, c'_v, d'_v) \ : \quad & S_{j-1} = c'_u S_{i-1} + d'_u S_i, \qquad S_j = c'_v S_{i-1} + d'_v S_i, \\ (c''_u, d''_u, c''_v, d''_v) \ : \quad & S_{k-1} = c''_u S_{j-1} + d''_u S_j, \qquad S_k = c''_v S_{j-1} + d''_v S_j, \end{aligned} \tag{3.2}$$

then (S_{k-1}, S_k) can be expressed by (S_{i-1}, S_i) as

$$S_{k-1} = c_u S_{i-1} + d_u S_i, \quad S_k = c_v S_{i-1} + d_v S_i, \tag{3.3}$$

$$\begin{cases} c_u = c''_u c'_u + d''_u c'_v, & d_u = c''_u d'_u + d''_u d'_v, \\ c_v = c''_v c'_u + d''_v c'_v, & d_v = c''_v d'_u + d''_v d'_v. \end{cases} \tag{3.4}$$

Let U and V be big-integers corresponding to two successive elements of the remainder sequence generated by M and S, hence $U > V \gg 1$. Let U and V be represented as $(u_m \neq 0)$

$$\begin{aligned} &U = u_m B^m + u_{m-1} B^{m-1} + \cdots + u_1 B + u_0, \qquad B > u_i \geq 0 \ \ (i = m, \ldots, 0), \\ &V = v_m B^m + v_{m-1} B^{m-1} + \cdots + v_1 B + v_0, \qquad B > v_i \geq 0 \ \ (i = m, \ldots, 0). \end{aligned} \tag{3.5}$$

We divide U and V into $m - \mu$ higher digits and $\mu - 1$ lower digits; the value of μ depends on the elimination step and will be specified below.

$$\begin{aligned} &U = \hat{U} B^\mu + \breve{U}, \quad \hat{U} = u_m B^{m-\mu} + \cdots + u_\mu, \quad \breve{U} = u_{\mu-1} B^{\mu-1} + \cdots + u_0, \\ &V = \hat{V} B^\mu + \breve{V}, \quad \hat{V} = v_m B^{m-\mu} + \cdots + v_\mu, \quad \breve{V} = v_{\mu-1} B^{\mu-1} + \cdots + v_0. \end{aligned} \tag{3.6}$$

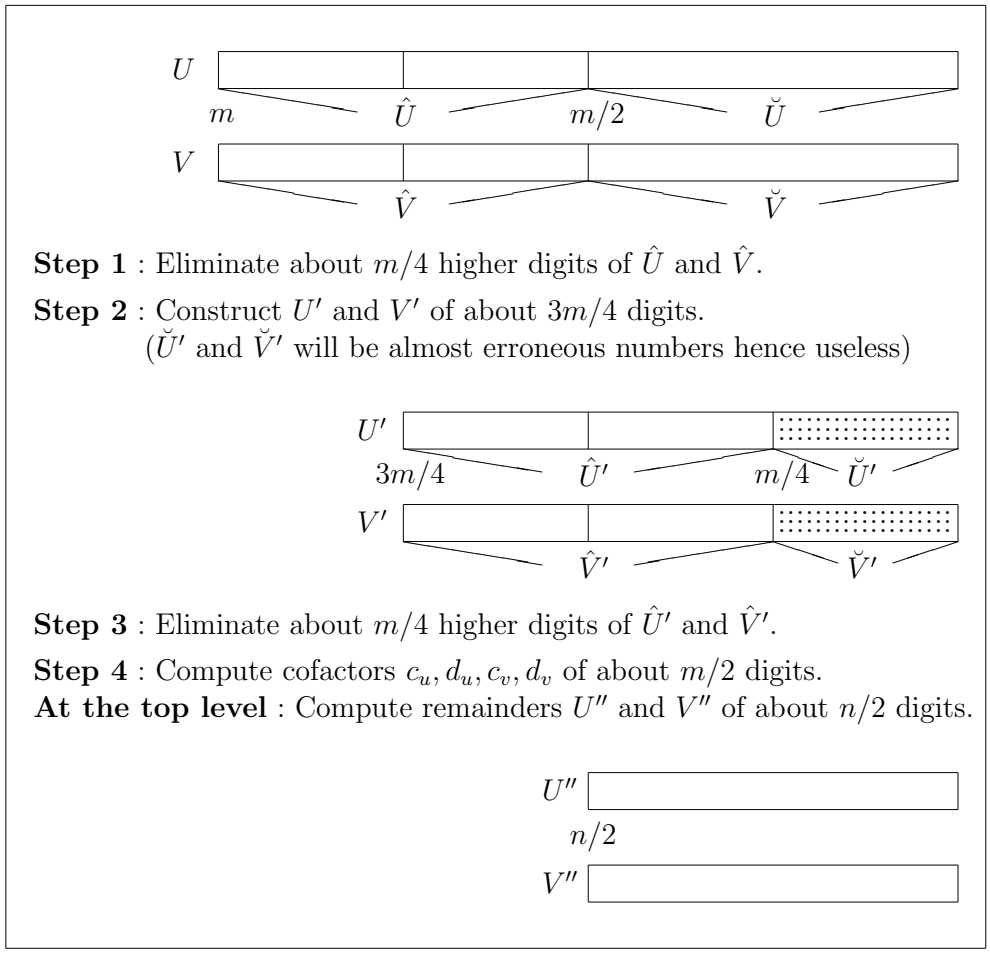


Figure 1: Illustration of our divide-and-conquer method

We perform the elimination of about $m/2$ higher digits of U and V by the following 4 steps, as illustrated by Figure 1.

recursive procedure Eliminate(U,V) ==
% Eliminate about $m/2$ higher digits of U and V, and
% return the cofactors c_u, d_u, c_v, d_v of about $m/2$ digits.
% At the top level, construct and return U'' and V''.

Step 1: Put $\mu \simeq m/2$ and divide U and V into $\hat{U}, \breve{U}$ and $\hat{V}, \breve{V}$, respectively, as in (3.6). Then, by calling **Eliminate** recursively, eliminate about $m/4$ higher digits of $\hat{U}$ and $\hat{V}$, and obtain cofactors c'_u, d'_u, c'_v, d'_v of about $m/4$ digits.

Step 2: Construct integers U' and V' of about $3m/4$ digits, as follows (U' and

V' correspond to S_{j-1} and S_j, respectively, in (3.2)).

$$U' = c'_u U + d'_u V, \quad V' = c'_v U + d'_v V. \tag{3.7}$$

Step 3: Put $\mu' \simeq m/4$ and divide U' and V' into $\hat{U}', \breve{U}'$ and $\hat{V}', \breve{V}'$, respectively, as in (3.6):

$$U' = \hat{U}' B^{\mu'} + \breve{U}', \qquad V' = \hat{V}' B^{\mu'} + \breve{V}'. \tag{3.8}$$

Then, by calling **Eliminate** recursively, eliminate about $m/4$ higher digits of $\hat{U}'$ and $\hat{V}'$, and obtain cofactors $c''_u, d''_u, c''_v, d''_v$ of about $m/4$ digits.

Step 4: Compute cofactors c_u, d_u, c_v, d_v of about $m/2$ digits, by formulas in (3.4). At the top level where $U = M$ and $V = S$, construct U'' and V'' as follows (U'' and V'' correspond to S_{k-1} and S_k, respectively, in (3.2)).

$$U'' = c_u M + d_u S, \qquad V'' = c_v M + d_v S. \tag{3.9}$$

□

Setting $U = M$ and $V = S$ initially, we see that the elimination of about $n/2$ higher digits of M and S is divided into two eliminations of about $n/4$ higher digits of four integers which are of about $n/2$ digits. Therefore, calling procedure **Eliminate** recursively, we can divide and conquer the elimination.

4. Details and verification of the method

In this section, we verify the divide-and-conquer method described in the previous section, by specifying details of the method. The problems we must solve are as follows.

Problem 1: The division of U and V shown in (3.6) is not trivial. For example, consider an extreme case that $\mathrm{digit}(U) > 2\,\mathrm{digit}(V)$. The division of V in (3.6), with $\mu \simeq m/2$, will give $\hat{V} = 0$. How do we treat such cases ?

Problem 2: There are many pairs of integers $\langle D', N' \rangle$ which satisfy $D'S \equiv N' \pmod{M}$. Among the pairs, we must find only one pair $\langle D, N \rangle$ that satisfies the conditions $0 < D < \sqrt{M/2}$ and $|N| < \sqrt{M/2}$. How do we find the required pair ?

Problem 3: If we are careless, the elimination by divide-and-conquering often gives results such that $|d_v| > \sqrt{M/2}$ at the top level of the elimination. How do we control the computation to avoid such cases ? Note that the upper equality in (2.8) gives $M = S_0 < 2|D_i|S_{i-1}$ because $|D_{i-1}| \leq |D_i|$ and $0 < S_i < S_{i-1}$, hence $|S_{i-1}| > M/(2|D_i|) > \sqrt{M}$ so long as $|D_i| < \sqrt{M}/2$.

Problem 4: In our divide-and-conquer method, we do not handle remainders $S_0, S_1, \cdots, S_\kappa$ but some higher digits of them. We must obtain the correct result even if we discard lower digits of $S_0, S_1, \cdots, S_\kappa$. How do we control the computation to do so ?

4.1. On Problems 1 and 2

We solve the Problem 1 as follows.

Abnormal case: The case that $\mathrm{digit}(U) - \mathrm{digit}(V) > \mathrm{digit}(U)/8$ is called *abnormal*.

Algorithm detail 1: In the abnormal case, we compute $W = \mathrm{rem}(U, V)$ and continue divide-and-conquering by setting $U := V$ and $V := W$. (If we still encounter the abnormal case, we continue the remainder computation.)

Algorithm detail 2: If $\mathrm{digit}(U) \leq 5$ then we stop divide-and-conquering and compute cofactors c_u, d_u, c_v, d_v by the extended Euclidean algorithm.

On Problem 2. The required pair $\langle D, N\rangle$ can be found in the remainder sequence $(S_1, S_2, \cdots, S_\kappa, \cdots)$ and the cofactor sequence $(D_1, D_2, \cdots, D_\kappa, \cdots)$. Let $(S_{\kappa-1}, S_\kappa)$ be a pair of successive elements of the remainder sequence, such that $S_\kappa = |N|$. In order to obtain the required pair $\langle D, N\rangle$, we must eliminate n' higher decimal digits of M, where n' is definite but unknown. We cannot perform such an elaborate elimination by the divide-and-conquering only. Therefore, we find the required pair as follows.

Algorithm detail 3: We use the divide-and-conquer method to obtain a pair $(S_{\kappa'-1}, S_{\kappa'})$, $\kappa' \leq \kappa$ hence $S_{\kappa'} \geq S_\kappa$, and find the required pair $(S_{\kappa-1}, S_\kappa)$ by computing the remainder sequence of $S_{\kappa'-1}$ and $S_{\kappa'}$. How to control the computation, see **details 4** and **5** given below.

4.2. On Problem 3

Problem 3 is very important actually. There are two ways to solve this problem. The first way is to use the fact that, given C_i, C_{i+1}, D_i and D_{i+1} for $i > 2$, we can compute C_{i-1} and D_{i-1} as

$$\begin{aligned} C_{i-1} &= \mathrm{rem}(|C_{i+1}|, |C_i|) \cdot \mathrm{sign}(C_{i+1}), \\ D_{i-1} &= \mathrm{rem}(|D_{i+1}|, |D_i|) \cdot \mathrm{sign}(D_{i+1}). \end{aligned} \tag{4.1}$$

Therefore, if we find that $|d_v| > \sqrt{M/2}$ then we back the cofactor sequences.

The second way is to stop the remainder sequence computation at the bottom level of divide-and-conquering as specified below. (2.7) tells us that

$$|d'_v| = \max\{|c'_u|, |d'_u|, |c'_v|, |d'_v|\}, \qquad |d''_v| = \max\{|c''_u|, |d''_u|, |c''_v|, |d''_v|\}. \tag{4.2}$$

Since (S_{i-1}, S_i) and (S_{j-1}, S_j) in (3.2) are pairs of two successive elements of a remainder sequence, (2.6) tells us that $c'_u c'_v < 0$, $d'_u d'_v < 0$, $c'_u d'_u < 0$, $c'_v d'_v < 0$, and similar inequalities for $c''_u, d''_u, c''_v, d''_v$. Therefore, in (3.4), we have

$$\begin{aligned} |c_u| &= |c''_u c'_u| + |d''_u c'_v|, \qquad |d_u| = |c''_u d'_u| + |d''_u d'_v|, \\ |c_v| &= |c''_v c'_u| + |d''_v c'_v|, \qquad |d_v| = |c''_v d'_u| + |d''_v d'_v|, \\ &\Longrightarrow\ |d_v| = \max\{|c_u|, |d_u|, |c_v|, |d_v|\} < 2|d'_v d''_v|. \end{aligned} \tag{4.3}$$

Let λ be the number of recursive calls of procedure **Eliminate**; $\lambda = 1$ at the top level of divide-and-conquering and $\lambda \simeq \log_2 |\mathrm{digit}(M/U)|$. We call λ *depth* of the divide-and-conquer. The remainder sequence computation at the bottom level must be stopped so that we will have $|d_v| < \sqrt{M/2}$ at the top level. If we set the stopping condition at the depth λ as $|d_v| = |d_k| < \sqrt[\lambda]{M/2} \le |d_{k+1}|$ then we will quite often have $|d_v| > \sqrt{M/2}$ at the top level. Therefore, we set the stopping condition as given below, then the last inequality in (4.3) assures that $|d_v| < \sqrt{M/2}$ at the top level.

Algorithm detail 4: At the bottom level of divide-and-conquering of the depth λ, where we compute the cofactor sequences $(d'_0, d'_1, d'_2, \cdots)$ and $(d''_0, d''_1, d''_2, \cdots)$ of $(U, V, U_2, \cdots)$ and $(U', V', U'_2, \cdots)$, respectively, by the extended Euclidean algorithm, we stop the remainder sequence computation and return cofactors $c'_{k-1}, d'_{k-1}, c'_k, d'_k$ etc., if we have

$$|d'_k| < \frac{(M/2)^{1/2^\lambda}}{2^\lambda} \le |d'_{k+1}|, \qquad |d''_k| < \frac{(M/2)^{1/2^\lambda}}{2^\lambda} \le |d''_{k+1}|. \tag{4.4}$$

4.3. On Problem 4

Suppose U and V are integers obtained by discarding lower $\ell-1$ digits of S_{i-1} and S_i, respectively. Then, we can express S_{i-1} and S_i as

$$\begin{aligned} S_{i-1} &= (U + \varepsilon_u) \times B^\ell, \quad 0 \le \varepsilon_u < 1, \\ S_i &= (V + \varepsilon_v) \times B^\ell, \quad 0 \le \varepsilon_v < 1. \end{aligned} \tag{4.5}$$

We call ε_u and ε_v *errors* of U and V, respectively. Similarly, suppose that U' and V' correspond to S_{j-1} and S_j, respectively, errors η_u, η_v of $\hat{U}, \hat{V}$ in (3.6) and errors η'_u, η'_v of $\hat{U}', \hat{V}'$ in (3.8) are defined by $(\nu = \ell + \mu)$

$$\begin{aligned} S_{i-1} &= (\hat{U} + \eta_u) \times B^\nu, \quad & S_{j-1} &= (\hat{U}' + \eta'_u) \times B^{\nu'}, \\ S_i &= (\hat{V} + \eta_v) \times B^\nu, & S_j &= (\hat{V}' + \eta'_v) \times B^{\nu'}. \end{aligned} \tag{4.6}$$

Remark 3: In (4.5) and (4.6), the errors are assumed to be less that 1. However, the errors will be accumulated during the computation, as will be explained later. Therefore, in the actual program, we treat ε_u and ε_v etc. as accumulated errors, hence they may be negative or greater than 1. □

By $(U_0 = S_{i-1}, U_1 = S_i, U_2, \cdots, U_l, \cdots)$ and $(\hat{U}_0 = \hat{U}, \hat{U}_1 = \hat{V}, \hat{U}_2, \cdots, \hat{U}_l, \cdots)$, we denote the remainder sequences generated by (S_{i-1}, S_i) and $(\hat{U}, \hat{V})$, respectively. The corresponding quotient sequences are denoted by

$$\begin{aligned} &(q_1, q_2, \cdots, q_l, \cdots), \quad && q_l = \mathrm{quo}(U_{l-1}, U_l) \ \ (l = 1, 2, \ldots), \\ &(\hat{q}_1, \hat{q}_2, \cdots, \hat{q}_l, \cdots), \quad && \hat{q}_l = \mathrm{quo}(\hat{U}_{l-1}, \hat{U}_l) \ \ (l = 1, 2, \ldots). \end{aligned} \tag{4.7}$$

We denote the cofactor sequences of $(\hat{U}_0, \hat{U}_1, \cdots, \hat{U}_l, \cdots)$ by $(c_0, c_1, \ldots, c_l, \ldots)$ and

$(d_0, d_1, \ldots, d_l, \ldots)$. (We omit $\hat{}$ for cofactor sequences.) Note that the sequences $(q_1, q_2, \cdots, q_{l-1}, \cdots)$ and $(U_2, U_3, \cdots, U_l, \cdots)$ are not computed.

In the rest of this subsection, we put $U = S_{i-1}$ and $V = S_i$ for simplicity, hence $U = (\hat{U} + \eta_u) \times B^\nu$ and $V = (\hat{V} + \eta_v) \times B^\nu$, instead of equalities in (4.6). Furthermore, we use indices i, j and k differently from the usage in Section 3.

We first consider necessary digits to assure that $\mathrm{quo}(U, V) = \mathrm{quo}(\hat{U}, \hat{V})$; the following proposition is used for computing the remainder correctly in the abnormal case.

PROPOSITION 1: *Let* $q = \mathrm{quo}(U, V)$ *and* $\hat{q} = \mathrm{quo}(\hat{U}, \hat{V})$. *We have* $q = \hat{q}$ *iff*

$$\hat{q}\eta_v - \eta_u \le \mathrm{rem}(\hat{U}, \hat{V}) < \hat{V} + (\hat{q} + 1)\eta_v - \eta_u. \tag{4.8}$$

Proof We have $\mathrm{rem}(U, V) = U - qV = [(\hat{U} - q\hat{V}) + \eta_u - q\eta_v]B^\nu = [(\hat{U} - \hat{q}\hat{V}) + \eta_u - \hat{q}\eta_v + (\hat{q} - q)(\hat{V} + \eta_v)]B^\nu$. Substituting the rightmost expression for $\mathrm{rem}(U, V)$ in inequality $0 \le \mathrm{rem}(U, V) < V = (\hat{V} + \eta_v) \times B^\nu$ and using $\mathrm{rem}(\hat{U}, \hat{V}) = \hat{U} - \hat{q}\hat{V}$, we obtain

$$0 \le \mathrm{rem}(\hat{U}, \hat{V}) + \eta_u - \hat{q}\eta_v + (\hat{q} - q)(\hat{V} + \eta_v) < \hat{V} + \eta_v. \tag{4.9}$$

If $\hat{q} = q$ then this inequality is identical to that in (4.8), hence $\langle q = \hat{q} \rangle \Longrightarrow$ (4.8). If $\hat{q} > q$ then the r.h.s. inequality in (4.9) gives $\mathrm{rem}(\hat{U}, \hat{V}) + \eta_u - \hat{q}\eta_v < 0$, contradicting (4.8). Similarly, if $\hat{q} < q$ then the l.h.s. inequality in (4.9) gives $\mathrm{rem}(\hat{U}, \hat{V}) + \eta_u - \hat{q}\eta_v \ge \hat{V} + \eta_v$, contradicting (4.8). Hence, (4.8) $\Longrightarrow \langle q = \hat{q} \rangle$. □

Next, we consider the normal case in which we can compute the remainder sequence $(\hat{U}_0 = \hat{U}, \hat{U}_1 = \hat{V}, \hat{U}_2, \cdots)$. We assume that the value of ν has been chosen so that $q_1 = \hat{q}_1$. Then, the quotient sequences in (4.7) will be such that, for some k, we have

$$q_i = \hat{q}_i \quad (i = 1, \cdots, k-1), \qquad q_k \ne \hat{q}_k. \tag{4.10}$$

The following proposition clarifies the relationship between k and η_u, η_v, and it is used to stop the computation at the bottom level of divide-and-conquering.

PROPOSITION 2: *The relations in* (4.10) *hold iff*

- *for any* j, $0 < j < k$:

$$-(c_{j+1}\eta_u + d_{j+1}\eta_v) \le \hat{U}_{j+1} < \hat{U}_j - (c_{j+1} - c_j)\eta_u - (d_{j+1} - d_j)\eta_v, \tag{4.11}$$

- *for* $j = k$:

$$\begin{aligned} \text{either} \quad & \hat{U}_{k+1} < -(c_{k+1}\eta_u + d_{k+1}\eta_v), \\ \text{or} \quad & \hat{U}_{k+1} \ge \hat{U}_k - (c_{k+1} - c_k)\eta_u - (d_{k+1} - d_k)\eta_v. \end{aligned} \tag{4.12}$$

Proof We have $q_1 = \hat{q}_1$. Assume that $q_i = \hat{q}_i$ $(i = 1, \ldots, j-1)$, then we have

$$c_i\hat{U} + d_i\hat{V} = \hat{U}_i, \quad c_iU + d_iV = U_i \qquad (i = 2, \ldots, j).$$

Hence, U_{j+1} can be expressed as follows.

$$\begin{aligned} U_{j+1} &= \mathrm{rem}(U_{j-1}, U_j) \;=\; (c_{j-1}U + d_{j-1}V) - q_j\,(c_jU + d_jV) \\ &= [(c_{j-1} - q_jc_j)\,(\hat{U} + \eta_u) + (d_{j-1} - q_jd_j)\,(\hat{V} + \eta_v)] \times B^\nu \\ &= [\hat{U}_{j-1} - q_j\hat{U}_j + (c_{j-1} - q_jc_j)\,\eta_u + (d_{j-1} - q_jd_j)\,\eta_v] \times B^\nu \\ &= [\hat{U}_{j-1} - \hat{q}_j\hat{U}_j + (c_{j-1} - \hat{q}_jc_j)\,\eta_u + (d_{j-1} - \hat{q}_jd_j)\,\eta_v] \times B^\nu \\ &\qquad + (\hat{q}_j - q_j)\,(\hat{U}_j + c_j\eta_u + d_j\eta_v) \times B^\nu. \end{aligned}$$

Since $\hat{U}_{j-1} - \hat{q}_j\hat{U}_j = \hat{U}_{j+1}$, $c_{j-1} - \hat{q}_jc_j = c_{j+1}$, $d_{j-1} - \hat{q}_jd_j = d_{j+1}$ and $\hat{U}_j + c_j\eta_u + d_j\eta_v = c_j(\hat{U} + \eta_u) + d_j(\hat{V} + \eta_v) = (c_jU + d_jV)/B^\nu = U_j/B^\nu$, we obtain

$$U_{j+1} = (\hat{U}_{j+1} + c_{j+1}\eta_u + d_{j+1}\eta_v) \times B^\nu + (\hat{q}_j - q_j)U_j.$$

Substituting the above r.h.s. expression for U_{j+1} in inequality $0 \le U_{j+1} < U_j$, we obtain

$$0 \;\le\; \hat{U}_{j+1} + c_{j+1}\eta_u + d_{j+1}\eta_v + (\hat{q}_j - q_j)U_j/B^\nu \;<\; U_j/B^\nu. \tag{4.13}$$

If $\hat{q}_j = q_j$ then this inequality is identical to that in (4.11), because $U_j = (\hat{U}_j + c_j\eta_u + d_j\eta_v)B^\nu$, hence $\langle q_j = \hat{q}_j\rangle \Longrightarrow$ (4.11). If $\hat{q}_j > q_j$ then the r.h.s. inequality in (4.13) gives $\hat{U}_{j+1} + c_{j+1}\eta_u + d_{j+1}\eta_v < 0$, or the upper inequality in (4.12) with $k = j$, contradicting (4.11). Similarly, if $\hat{q}_j < q_j$ then the l.h.s. inequality in (4.13) gives $\hat{U}_{j+1} + c_{j+1}\eta_u + d_{j+1}\eta_v \ge \hat{U}_j + c_j\eta_u + d_j\eta_v$, or the lower inequality in (4.12) with $k = j$, contradicting (4.11). Hence, (4.11) $\Longrightarrow \langle q_j = \hat{q}_j\rangle$. Therefore, we obtain Proposition 2 because $\langle \neg$ (4.11) for $j = k\rangle =$ (4.12). □

With Proposition 2, we control the computation as follows, then we obtain the correct cofactors by handling truncated integers with errors.

Algorithm detail 5: At the bottom level of divide-and-conquering, where we compute the cofactors by the extended Euclidean algorithm, we stop the computation of remainder sequence if the inequality in (4.11) becomes unsatisfied (and inequality for $\hat{U}'_j$ and $\hat{U}'_{j+1}$, too).

4.4. Propagation of errors

Let us finally consider the propagation of errors. At the top level of divide-and-conquering, we have $U = M$ and $V = S$ hence $\varepsilon_u = \varepsilon_v = 0$. Suppose, at a lower level, U and V have errors ε_u and ε_v, respectively. These errors affect the errors η_u and η_v in (4.6) only a very little; in fact, by (4.5) and (4.6) we obtain

$$\eta_u = (\breve{U} + \varepsilon_u)/B^\mu, \qquad \eta_v = (\breve{V} + \varepsilon_v)/B^\mu. \tag{4.14}$$

In Step 2, c'_u, c'_v and d'_u, d'_v are multiplied to U and V, respectively, which magnifies the errors of U and V by amounts of $O(B^{m/4})$ in U' and V'. Thus, $\breve{U}'$ and $\breve{V}'$ will be almost erroneous numbers. However, the errors η'_u and η'_v in (4.6) are not large; they are given by

$$\eta'_u = (\breve{U}' + c'_u\varepsilon_u + d'_u\varepsilon_v)/B^{\mu'}, \qquad \eta'_v = (\breve{V}' + c'_v\varepsilon_u + d'_v\varepsilon_v)/B^{\mu'}. \tag{4.15}$$

Note that η'_u and η'_v may be negative because $c'_u d'_u < 0$ and $c'_v d'_v < 0$. If we are not at the bottom level, we must still divide-and-conquer $(\hat{U}, \hat{V})$ and $(\hat{U}', \hat{V}')$, by resetting $(U, V, \varepsilon_u, \varepsilon_v)$ as

$$(U, V, \varepsilon_u, \varepsilon_v) \ := \ (\hat{U}, \hat{V}, \eta_u, \eta_v), \qquad (U, V, \varepsilon_u, \varepsilon_v) \ := \ (\hat{U}', \hat{V}', \eta'_u, \eta'_v). \tag{4.16}$$

Actually, the conditions in **detail 4** are stronger than those in **detail 5** for stopping the remainder sequence computation. Therefore, we may represent the errors by single precision numbers.

Algorithm detail 6: We represent the errors by interval numbers.

5. Implementation and experiments

We have implemented our divide-and-conquer method on NS-Lisp (Nara Standard Lisp implemented by Kako, Nara Women's University, Japan) which is equipped with Karatsuba's method for the multiplication of big-integers; for Karatsuba's method, see [GG99], Sec. 8.1. We represent $M, S, U, V, U', V', U'', V''$ etc. by user-defined lists; for example, $U = u_m B^m + u_{m-1}B^{m-1} + \cdots + u_0$ is represented by a list $((m, \varepsilon_u), u_m, u_{m-1}, \cdots, u_0)$. On the other hand, we treat cofactors c'_u, d'_u, c'_v, d'_v etc. as Lisp numbers, and Karatsuba's method is applied to the multiplication of them. The multiplications $c'_u \times U$ etc. are done by Karatsuba's method by converting U etc. to Lisp numbers. We set B in two ways as

$$B = 10^8 \quad \text{and} \quad B = 2^{24}. \tag{5.1}$$

For comparison, we have also implemented Wang's algorithm (extended Euclidean method) and its enhanced version using Lehmer's technique. We represent S_i as well as M and S by user-defined lists as explained above, while Q_i, C_i and D_i by Lisp numbers. The multiplications $Q_i \times S_i$ etc. are done by the classical method by preserving the list representation of S_i.

Table I shows the result of experiments on Pentium II (300MHz), where M and S were generated randomly satisfying $\mathrm{digit}(M) = \mathrm{digit}(S)$. (Note that "digit 1000", for example, in the table means that M and S are integers of about 8000 decimal digits.) Although our current program is in a crude level, the result shows clearly the superiority of the divide-and-conquer method.

Remark 4: In (3.7), only about higher $m/2$ digits are necessary in the subsequent computation. Therefore, we can speed up the computation in Step

	Case of $B = 10^8$ (sec)			Case of $B = 2^{24}$ (sec)		
digit	Euclid	Lehmer	DivConq	Euclid	Lehmer	DivConq
100	0.896	0.212	0.0530	0.778	0.213	0.0540
200	3.41	0.778	0.129	2.94	0.783	0.124
300	7.66	1.70	0.220	6.56	1.71	0.21
400	13.5	2.98	0.332	11.6	3.01	0.311
500	21.1	4.62	0.465	17.6	4.65	0.438
600	30.0	6.64	0.595	26.0	6.69	0.561
700	40.8	8.99	0.773	35.3	9.02	0.724
800	53.5	11.7	0.927	46.0	11.8	0.863
900	67.3	14.8	1.12	58.5	14.9	1.04
1000	83.2	18.2	1.34	71.7	18.3	1.26

Table 1: Result of experiments

2 considerably. However, neither our implementation nor the following time-complexity analysis takes this speed-up into account. □

We give the time-complexity of the divide-and-conquer method. By $M(m, m')$ we denote the time for multiplying two integers of m and m' digits. By $T(n)$ and $E(n)$, we denote the total computation time and the cost of elimination, respectively, for n digit integers. Then, we have

$$\begin{aligned} T(n) &= 4M(n/2, n) + E(n), \\ E(m) &= 2E(m/2) + 4M(m/4, m) + 8M(m/4, m/4). \end{aligned} \tag{5.2}$$

Here, $4M(n/2, n)$ is the cost of multiplications in (3.9), and $8M(m/4, m/4)$ is the cost of computation of c_u, d_u, c_v and d_v in Step 4. Bounding $M(n/2, n)$ and $M(m/4, m)$ as $M(n/2, n) \leq 2M(n/2, n/2)$ and $M(m/4, m) \leq 4M(m/4, m/4)$, we can estimate $T(n)$ as follows.

$$T(n) \ \leq \ 8M(n/2, n/2) + nE(1) + 6 \sum_{k=2}^{\log_2 n} 2^k M(n/2^k, n/2^k). \tag{5.3}$$

For Karatsuba's method, we have $M(m, m) \approx 9m^{1.59}$ hence $E(n) \leq 95n^{1.59}$.

Let us briefly explain why the divide-and-conquer method is fast. In the extended Euclidean algorithm, we compute a quotient $Q_i = \mathrm{quo}(S_{i-1}, S_i)$ at each step, and Q_i is multiplied to S_i, C_i and D_i which are big integers during the most stage of computation. Q_i is usually a small number of one or several decimal digits, hence the fast multiplication algorithms are powerless for computing $Q_i S_i$, $Q_i C_i$ and $Q_i D_i$. On the other hand, the most time-consuming operations in the divide-and-conquer method are multiplications in (3.4), (3.7) and (3.9). The multiplicands in these cases are big integers, and we can apply the fast multiplication algorithms.

References

[GG99] J. von zur Gathen and J. Gerhard: *Modern Computer Algebra*. Cambridge University Press 1999.

[Knu69] D. E. Knuth: *The Art of Computer Programming*, Vol. 2 (Seminumerical Algorithms). Addison Wesley, 1969, Section 4.5.

[Moe73] R. Moenck: Fast computation of GCDs. *Proceedings of 5th ACM Annual Symposium on Theory of Computing*, pp. 142–171, ACM Press, New York, 1973.

[PW02] V. Y. Pan and X. Wang: Acceleration of Euclidean algorithm and extensions. *Proceedings of 2002 International Symposium on Symbolic and Algebraic Computation*, pp. 207–213, ACM Press, New York, 2002.

[Sch71] A. Schönhage: Schnelle Berechnung von Kettenbruchentwicklungen. Acta Informatica Vol. 1, 1971, pp. 139–144.

[Sor94] J. Sorenson: Two fast GCD algorithms. J. Algorithms, Vol. 16, 1994, pp. 110-114.

[SS92] T. Sasaki and M. Sasaki: On integer-to-rational conversion algorithm. SIGSAM Bulletin, Vol. 26, ACM, 1992, pp. 19–21.

[ST89] T. Sasaki and T. Takeshima: A modular method for Gröbner-basis computation over $\mathbf{Q}$ and solving system of algebraic equations. J. Inf. Proces., Vol. 12, 1989, pp. 371–379.

[STS01] A preliminary version of the present paper was presented at *RIMS Symposium on Theory and Application of Formula Manipulation* (University of Kyoto, Japan), November, 2001.

[Wan81] P. S. Wang: A p-adic algorithm for univariate partial fractions. *Proceedings of ACM Symposium on Symbolic and Algebraic Computation*, pp. 212–217, ACM, 1981.

[WGD82] P. S. Wang, M. J. T. Guy and J. H. Davemport: P-adic reconstruction of rational numbers. SIGSAM Bulletin, Vol. 16, ACM, 1982, pp. 2–3.

[Win88] F. Winkler: p-adic methods for the computation of Gröbner bases. J. Symb. Comput., Vol. 6, 1988, pp. 287-304.

Syzygies, and the Stabilization of the Numerical Buchberger Algorithm*

Carlo Traverso

Dipartimento di Matematica, Via Buonarroti 2, I-56127 PISA
traverso@dm.unipi.it

Abstract

We give two variants of Buchberger algorithm to compute an approximate Gröbner basis of an unstable set of polynomials given approximately. Both algorithms use syzygies in a different way: the first is a revisiting of the classical trace lifting algorithm in a generalized context, and can be used when the problem is generic in an explicitly given family; the second can be used in general, and uses intervals to discover possible vanishing and optimization techniques to validate a possible zero identified by interval arithmetic.

The first algorithm has been completely implemented and tested, the second is still in the preliminary design and experimenting phase.

KEYWORDS: Gröbner basis, floating point arithmetic, unstable approximate systems

1. Introduction

In (Traverso and Zanoni, 2002) we have discussed the possibility of using some forms of floating point arithmetics to perform a Gröbner basis computation through Buchberger algorithm, and shown in all the test cases that the result coincides (up to some approximation) with the result obtained using integer or rational arithmetic, but with a lower cost for the arithmetic computations, provided that the precision of the floating point arithmetic is sufficient; and otherwise the algorithm exhibits the complete loss of precision, and exits with an error condition. The result computed is an approximation to the result computed with exact arithmetic, even when the original problem is unstable.

When the input data are inexact, this is sometimes not what is sought; we

*This work was supported by the projects "Tecniche per Immagini", cluster C15, progetto n. 7 "Sviluppo, Analisi ed Implementazione di Metodi Matematici Avanzati per il Trattamento di Immagini" and "Algebra Commutativa e Computazionale", PIN-2001.

compute an approximation of the solution of a problem that is "near" to the "true" problem, but the solution of the nearby problem may be very far from the solution of the original problem; an example is when the original problem is an overdetermined system of equations (one with more equations than unknowns) in which a solution exists for physical reasons, and the coefficients are experimentally (hence inexactly) determined.

In this case, one has to use some additional information (e.g. the physical existence of a solution) to recover an approximation of the "exact" result.

When a qualitative information of this kind is not available, one is usually interested in finding an approximation of a solution of the "most degenerate" problem compatible with the data; for example, finding an approximate GCD one is interested in the highest degree such GCD compatible with the bounds.

We describe in this paper two different approaches, both of them based on the consideration of syzygies.

The first approach uses the syzygies of a different problem generic in a family containing the problem of which our data are an approximation to deduce syzygies for our computation. This approach requires the knowledge of some data, in particular the irreducibility of the family, and the genericity of our original problem. Otherwise it may fail.

This first approach generalizes the well-known *modular trace lifting Buchberger algorithm*, see (Traverso, 1988), that is a special case of this algorithm. Our generalized algorithm has been implemented, and works remarkably well on some important practical tests. The algorithm may fail if the problem under consideration is not generic (and in this case the algorithm detects the non-genericity), and may also fail if the data for the pilot computation are not generic but only random, like in the case of the trace-lifting algorithm, when an unlucky prime is chosen. This is however unlikely to happen.

The second approach uses interval arithmetic and the *extended Buchberger algorithm*, see (Caboara and Traverso, 1998), to deduce interval equations, and a subsequent step allows to validate either the vanishing of a solution in an interval of confidence (hence deducing a non-generic syzygy) or proving that such a solution does not exist, hence refining some computed intervals.

This last step can be solved either through quadratic programming, or through an ad-hoc iterative procedure; this part is however up to now theoretical only, and experiments have to be conducted to test the practical behaviour of this algorithm. A first full implementation is planned, although most components needed to test this approach are available.

To our knowledge, the only prior results in this domain are contained in (Gianni et al., 1998); their approach is to reduce the problem to a linear algebra problem and to solve a problem of singular values; the approach is however only possible for zero-dimensional ideals. Our approach is instead fully general, and makes no assumption on the geometrical properties of the result.

The sections 2 and 3 contain a review of previous results, some of them of

general type but possibly not known to the audience of this paper, and some of previously published research of the same author. The original part is contained in the following sections. Some of the ideas of this paper have been anticipated in (Traverso and Zanoni, 2002) (more in the conference than in the proceedings) and (Traverso, 2002), but both the implementation and the formalization as generalized trace lifting algorithm and the reduction of zero-decision ad refinement of an interval to a quadratic optimization problem are new.

2. Families of ideals and Buchberger algorithm

We begin with some notation; a product of variables is called a *power product* (PP for short); we assume to have a term-ordering, i.e. a noetherian total ordering on the set of power products; if f is a polynomial, it has a *leading coefficient* $Lc(f)$ and a *leading PP* $Lpp(f)$; to avoid to treat the zero polynomial in a special way, we define a special PP, *NAPP* (*NotAPowerProduct* in analogy with NAN, *NotANumber*; hence $Lc(0) = NAN$, $Lpp(0) = NAPP$. *NAPP* is considered to be smaller than any other PP.

For the purpose of Buchberger algorithm, an ideal is represented by a set of generators (a basis), consisting in a set of polynomials in a polynomial ring $k[X]$ over the ground field k. An ideal $I \subseteq k[X]$ together with a basis (f_i) is called a *based ideal*. Buchberger algorithm transforms a based ideal into an *equivalent Gröbner-based ideal*, for obvious definitions of what is undefined above.

Such a structure allows a parametric representation: consider an ideal I generated by a set of polynomials f_i in $R[X]$, R being a ring, and let $\mathbf{p}$ be a geometric point (a map $R \to K$ into a field K, the map being denoted as $\mathbf{p} : a \mapsto a(\mathbf{p})$); then we have and ideal $I(\mathbf{p})$ in $K[X]$ generated by $f_i(\mathbf{p})$, the map $\mathbf{p}$ being extended trivially to a map $R[X] \to K[X]$.

Special cases are $R = \mathbf{Z}$ (and in this case the points are, substantially, $\mathbf{Q}$ and the various $\mathbf{Z}_p$) and $R = k[t_i]$, and giving a point consists in giving a value to the parameters t_i. Here *substantially* means *up to an extension of the ground field*, that does not affect Buchberger algorithm. In this viewpoint, a based ideal of $R[X]$ is seen as a *family* of based ideals on $\mathrm{Spec}(R)$.

When I is a based ideal with coefficients in a field k, we can embed it in a family replacing every non-leading non-zero coefficient with a different invertible indeterminate: if $f_i = \sum_J a_{i,J} X^J$ consider $R = k[t_{i,J}, t_{i,J}^{-1}]$ and $\bar{I} = (\bar{f}_i = \sum_J t_{i,J} X^J)$. We call this the *universal family associated to* I.

We want to study the behaviour of Buchberger algorithm, and in particular the Gröbner basis, when $\mathbf{p}$ varies. This *behaviour* can be formalized through traces.

We assume for simplicity that the input polynomials of Buchberger algorithm are uniquely identified by their *Lpp*, and that Buchberger algorithm generates polynomials that have different *Lpp* (this second assumption is true for the classical Buchberger algorithm, but may be false for variants); if this assumption

is false, one has to modify slightly the definition, using instead of PPs unique identifiers for polynomials that contain their *Lpp*.

A *Buchberger trace*[†] is a list of triples of PP. To each critical pair considered in Buchberger algorithm we associate a triple consisting of the *Lpp* of the polynomials of the pair, and the *Lpp* of the polynomial that is the result of the simplification procedure. (the third element of a triple can hence be a *NAPP*, and not a "true" PP). A Buchberger trace is a summary of the Buchberger algorithm,

Since Buchberger algorithm is an algorithm scheme depending on strategies, we assume that the strategies do not depend on coefficient values, as long as they are not zero; this is restrictive, since stability considerations for floating point computations might require that strategies consider coefficients, but on the other side floats are not even a ring. To simplify some statements, we even assume that the strategies only depend on the leading terms of the polynomials involved in the algorithm. This is sometimes not true for some implementations, that might consider polynomial lengths, but is true in first approximation.

It is well known, and anyway trivial, that the behaviour of Buchberger algorithm in a family is constant on a (Zariski) open subset of the family; the reason is that the branching points of the algorithm depend on the vanishing of coefficients, and coefficients during the algorithm are rational functions of the input coefficients; hence the behaviour is constant on the open set where no coefficient vanishes that does not vanish everywhere. (If the ring R has no zero divisor this open set is non empty, otherwise one can slightly modify the argument). On this open set, we say that the Buchberger algorithm has *generic behaviour*.

One can subdivide a family into locally closed subfamilies (intersection of an open and a closed subfamily) in each of whose the behaviour of Buchberger algorithm is constant. Such a subdivision is called a *Buchberger stratification*. Remark that it may not be a stratification, in technical terms, since it is possible that the closure of a stratum meets another stratum without containing it.

If Buchberger algorithm has the same behaviour in two different points, the staircase of the Gröbner basis is the same at the two points; the converse is of course false. The subdivision of the parameter space induced by the staircase is not a stratification (and its elements are not even locally closed, but only constructible, i.e. finite union of locally closed subsets).

3. Floating point arithmetics for Buchberger algorithm

In this section we recall the essential points of (Traverso and Zanoni, 2002).

When handling inexact arithmetics inside of algebraic algorithms, the main problem is the handling of equality, in particular the equality to zero, i.e. the zero test. When we have to decide if a coefficient of a polynomial is zero, we

[†]in (Traverso, 1988) this was called a *Gröbner trace*, but the name *Buchberger trace* seems more adequate, since the trace depends on the special instance of Buchberger algorithm that is chosen

can have have absolute tests, context tests (when we compare a coefficient with the other coefficients) and historical tests (when we compare a coefficient with the coefficients that have originated it). And the result of a zero test is not a simple yes/no, but may allow a "maybe", that requires further action, that may depend on the context. This action might include raising an error condition that makes the algorithm fail. Defining an arithmetic consists not only in specifying the arithmetic operations, but also in specifying a zero test; this is also needed in inverting a number, that is possible only after a zero test with negative result. We say that a number *may be zero* if the zero test returns either "true" or "maybe".

Another important issue is to decide if the accumulated error has made a number completely unreliable, or more generally to estimate the residual precision.

A final issue is that we need to map a dense subset of the rationals into the floats, and this map is an "approximate homomorphism" in some sense that we do not need here to make more precise.

Pure floating arithmetic[‡] is unable to handle these issues, hence we have designed and tested three different arithmetics, that can be combined, and that are useful in different contexts.

Hybrids

The first is the *Hybrid arithmetic*, in which a number is composed of a float and an integer modulo a prime p; the two parts correspond to two different approximations of a rational, hence if the modular part is not zero then the number is not zero; if it is zero then the number is surely non invertible, but the zero test is positive only if it is corroborated by another context information; either the number originates from a difference of almost equal numbers (more explicitly, it is a result of an operation $a = r + s$, and the absolute value $|a| \leq \epsilon_1 r$ for a predetermined small positive constant ϵ_1), or a is a coefficient of a polynomial $\sum a_I X^I$ and $a \leq \epsilon_2 a_I$ for each a_I whose modular part is non zero. This last test includes the condition that a polynomial that is identically zero mod p is zero.

A map from a rational to a hybrid is obvious if no denominator is multiple of p; a map to a float to an hybrid is possible, starting from any rational approximation of he float; however the mod p part is in this case completely random, since in every intervals of the reals exist rationals having any residue mod p. Hence, substantially, one maps floats to hybrids assigning randomly the modular part.

Hybrid arithmetics does not allow to check the residual precision, but allows reduction mod p; hence for them a classical trace lifting algorithm is possible.

[‡]Under *floats* we design not only machine floats as the types `float` and `double`, but also bigfloats like e.g. **gmp** bigfloats, (Granlund, 1991-2002).

Double-floats

A second arithmetics is the *double-float arithmetic*; a number is represented by two floats with different length, e.g. a `float` and a `double`, a *short* and a *long* component.

A double-float allows to check the loss of precision, being the number of trailing bits of the short that are different from the corresponding bits of the long, since we expect the error to be random, and to influence approximately the same number of bits of the short and of the long, hence we expect that the first dirty bits of the short are exact in the long. It allows a zero-test, that is true when the absolute value of the short is larger than a suitable multiple of the long. To understand this, remark that a zero can only be obtained through cancellation in a sum; in this case, the floats are not "exactly zero", but have a "noise" that is, in base 2 logarithm, of size equal to the number of inexact bits minus the number of total bits of the float; we may expect that the influence of error accumulation is approximately the same on the long and on the short, hence when we compute inexactly a zero, we expect that the short is about 2^d times larger than the long, where d is the difference in bits between the long and the short.

Double-floats allow to deduce the precision of the computed result, through the number of common bits of the two components (equivalently—almost...—through the binary logarithm of the quotient of the first component by the difference of the two).

Intervals

The third arithmetic is interval arithmetic: a number is represented by a pair of numbers, representing an interval, and arithmetic operations are performed in a way that represents elementwise operations on sets. Hence a number may be zero if zero is in the interval, but there is no test to decide zero equality.

Interval arithmetic can be used stating that a number is zero if and only if it may be zero, but as far as Buchberger algorithm is concerned this is of little use. since error accumulation usually leads to overestimating the zero test, unless the input precision of the initial intervals is much larger than the usual data precision. See the section 6 for further reasons why pure interval arithmetic is unsuitable for Buchberger algorithm.

Composite arithmetics

It is possible to combine features of different arithmetics: we can for example consider the *double hybrids* that combine a double float with an integer mod p (or equivalently an hybrid based on a double-float arithmetic).

Other possibilities are *multi-hybrids*, combining several integers mod p_i with a float or a double-float, *pointed-intervals* that combine an interval and a double-float inside it, and the "grand-total" *hybrid-pointed-intervals* combining a pointed-interval and an integer mod p (or several ones...).

For these mixed arithmetics it is necessary to specify which action we have

to take when the different subcomponents give discording results on some zero test; for example, when the interval and the modular part give discording values of the may-be-zero test. This may be algorithm-dependent.

4. Trace lifting revisited

The trace lifting algorithm defined in Traverso (1988) computes a Gröbner basis for an ideal generated by a basis with integer coefficients, first performing a pilot computation of the ideal obtained reducing the basis modulo a prime p, keeping a Buchberger trace; this trace is used in the Buchberger algorithm for the original ideal, to flag "useless" pairs to be discarded. The third element in a triple is used to check that the result of the simplification with integer (rational) coefficients has the expected PP. The algorithm fails when this test fails, and might give wrong results if the prime chosen is unlucky.

The trace lifting algorithm can be described in the setting of families. Since the original basis has integer coefficients, we consider $\mathbf{Z}$ as ground ring, and consider Spec $\mathbf{Z}$ as basis of the family; the geometrical points of Z are $\mathbf{Q}$ (that is a generic point, i.e. its Zariski closure coincides with Spec $\mathbf{Z}$) and $\mathbf{Z}_p$; the Buchberger stratification consists of a generic component, containing $\mathbf{Q}$ and all the $\mathbf{Z}_p$ with p "lucky", and some components composed each of a finite number of "equally unlucky" primes. The trace lifting algorithm tries to compute the Buchberger trace of the ideal over $\mathbf{Q}$ computing the Buchberger trace of the ideal over a random prime, hoping not to hit an "unlucky" prime, that makes the algorithm either fail or to give a wrong result.

We can apply the same ideas to more general families, and to the case of floating arithmetic (when the coefficient ring is only approximately a ring, the coefficients are imperfectly known, the zero test may return "maybe"), and moreover we know that the input is in the family, but we do not know for sure that it is a generic element of the family. This last generalization corresponds, in the classical integer arithmetic case, to use the Buchberger trace computed mod p to compute the Gröbner basis modulo a different prime q that cannot be assumed to be lucky.

More precisely, we use the trace computed for a member of a family with an exact arithmetic (usually, on a point that is a map to a finite field) both as an hint to discard useless pairs and as an hint to decide equality to zero when the arithmetic test answers "maybe".

The algorithm can be used in two flavors; in one we discard a pair if the trace says that it is useless, in the other the pair is not discarded, and we check that the pair is indeed useless; the trace is only used to decide zero test in uncertain cases.

The algorithm pre-computes the trace, then performs a loop on it as follows: the Gröbner basis is accumulated in G, and we assume that an element of G is uniquely identified by its PP (otherwise the standard modifications are applied);

we assume that every element of G has a leading coefficient such that the zero-test returns "False"; let (PP_1, PP_2, PP_3) be the trace element under consideration:

- (optional step): If $PP_3 = NAPP$ do nothing and return; otherwise
- Take $f_1, f_2 \in G$ having PP_1, PP_2 as leading PP; compute their S-polynomial, and reduce it through the elements of G, obtaining g.
- If the leading PP of g is larger than PP_3, and its coefficient is maybe-zero, erase the leading term and repeat.
- If the leading PP of g is larger than PP_3 and surely not zero, report a failure;
- If the leading coefficient of g is maybe-zero, or the leading PP is smaller than PP_3, report a failure;
- otherwise, add g to the basis and continue the loop.

The failures are hence of three types; we have different diagnoses and recoveries possible:

1. the leading PP anticipated by the trace is smaller than the computed one:
   (a) the pilot computation has been performed at a non-generic (unlucky) point; one can recompute a different pilot computation;
   (b) the floating computation experienced a loss of precision mistaking a zero for a surely non zero element.
2. the leading PP anticipated by the trace is larger than the computed one:
   (a) the original problem is not generic;
   (b) the floating computation experienced a loss of precision mistaking a non zero element for a zero.
3. the leading coefficient is maybe zero.
   (a) the original problem is not generic;
   (b) the floating computation experienced a loss of precision mistaking a non zero element for a zero;
   (c) the indetermination of the floating coefficients is such that different traces are possible, one has to decide which branch to follow.

The failures 1b, 2b, 3b can be detected with an arithmetic controlling the loss of precision, e.g. through double-floats; 1a requires to redo the pilot computation; 2a, 3a and 3c require to abandon the trace algorithm, and determine (if possible) a new family, or resort to completely new ideas, like those exposed in section 6.

Remark that if during the Buchberger algorithm one discovers a potential basis element whose leading coefficient is maybe zero, there is the possibility of delaying the decision (i.e. put aside the polynomial, hoping that later another basis element with a sound leading coefficient can erase it), but otherwise eventually the issue has to be decided, either on the zero or on the non-zero side,

possibly forking the computation; indeed, in the original Buchberger algorithm one has to divide by the leading coefficient (that cannot be invertible if it may be zero), and in a denominator-free version allowing such a leading coefficient in a basis element may cause completely wrong results.

5. Overdetermined systems

A very special case, but quite important for applications, is the case of an overdetermined system, with more equations than unknowns, in which we know that one root exists. Such systems of equations appear quite frequently, for example in vision and computer aided design; such systems do not appear frequently in test collections related to symbolic computation, since they indeed cannot be handled symbolically. The usual symbolic recipe is to solve an equidetermined subsystem identifying all the roots, then select the "best" one; the numerical recipe instead is to add variables and solve a minimum problem.

We can embed our system in the family of all systems having the same support and at least one root; if all the equations have a constant term, this family is an irreducible one, since given an explicit root any system can be moved to one having that root changing the constant. It is easy to find a generic element in this family with coefficients modulo a prime p.

It is still possible that our algorithm fails, for example if the system has more roots, but in any case in which a result is obtained then the result is correct, (if the hypothesis of the existence of a root is correct) and can be checked checking the root. The root will be approximate, and can be a good starting point of an iterative numerical solving method.

One such example is the **kruppa** example in (Mourrain, 1996-2002), being a system of 6 equations in 5 unknowns. Our algorithm is better than solving a subsystem, since it involves fewer polynomials and of lower degree, and uses floating arithmetic instead of a costly exact arithmetic.

6. Special approximate systems and interval arithmetic

Sometimes it may be that we cannot identify a family of which our problem, imperfectly specified, is a generic element. In that case it is usually important to find the "most degenerate" member of a family that satisfies a set of initial constraints. This means that whenever a leading coefficient of a polynomial may be zero (i.e. its being zero is compatible with the initial constraints) we have to consider the coefficient as being zero, and add this constraint to the set of constraints. Remark that this possibility may be indicated by an answer "maybe" to the zero test, but the arithmetics may give "false maybe" because of different types of error accumulation.

The initial constraints are usually of the type of an interval in which each input coefficient lies; it is possible that some input coefficients are exact, while

others are variable in an interval, our analysis will apply to both types, and other types of constraints too.

In this case, some form of interval arithmetic is needed. In our analysis and preliminary experiments we have used coefficients composed of a double-float and an interval enclosing it. The double-float (the *central value*) is used to give an answer "yes" to the zero test, and the interval (the *confidence interval*) is used to give the answer "maybe" to the zero test; if both fail, the answer is "no". To avoid problems with the use of floating arithmetic for the interval ends, in the case of exact coefficients we take intervals that are very small, but wider than the smallest interval representable in the arithmetic used.

This said, interval arithmetic is subject to error accumulation, mainly due to the fact that the associative property is not true: if A, B, C are intervals, $A(B + C) \subseteq AB + AC$, and equality is false when B and C have opposite signs (an interval is positive, or negative, when both endpoints are such; otherwise its sign is "maybe zero").

We show how we can reduce the effect of error accumulation, through the use of syzygies, and how we can validate a "maybe zero" test.

Instead of Buchberger algorithm, we use the *extended Buchberger* algorithm, (Caboara and Traverso, 1998), that consists in the following: instead of considering $f_1, \ldots, f_n \in k[X]$ we consider vectors $(f_1, 1, 0, \ldots, 0), \ldots (f_n, 0, \ldots, 0, 1) \in k[X]^{n+1}$, and in the $k[X]$-module $k[X]^{n+1}$ we consider a term-ordering in which any PP in initial position is larger than any PP in any other position (for modules, the term ordering is defined on pairs (PP,position)).

In this way, exactly as in the extended euclidean algorithm, every polynomial computed during Buchberger algorithm is supplemented by a generation, i.e. a representation $g = \sum \phi_i f_i$; the different representations of 0 obtained are the syzygies with respect to the input basis, but we are rather interested to the other elements, in which we represent a non-zero element g; indeed, we are interested in the case in which the leading coefficient is maybe zero (as interval).

Assume now that $f_i = \sum a_{i,\alpha} X^\alpha$, $\phi_i = \sum b_{i,\beta} X^\beta$, hence $g = \sum c_\gamma X^\gamma$, where

$$c_\gamma = \sum_{i,\alpha+\beta=\gamma} a_{i,\alpha} b_{i,\beta}. \tag{1}$$

We now regard equations (1) for varying γ as a system of linear equations, in the indeterminates $c_\gamma, b_{i,\beta}$; the system is homogeneous, but some of the $b_{i,\beta}$ can be exactly determined as being non zero (coming from S-polynomials), and some of the c_γ are exactly determined as 0.

Of this system we have an approximate solution as intervals, and the highest of the non-zero c_γ (let it be c) is an interval containing zero.

The first step is hence to refine this solution, through interval linear algebra; in general, the confidence intervals will be narrowed, and it may happen that c has now a determined sign, hence the "maybe zero" test is resolved to the negative. The Buchberger algorithm can go on, with the refined confidence intervals.

There is however a second problem: the coefficient matrix has a Toeplitz-like structure, since although every $a_{i,\alpha}$ is an interval, every $a_{i,\alpha}$ in every equation $c_\gamma = \ldots$ has the same exact real value. It may be that with this further constraint on the coefficients the realizable interval for c_γ is smaller than what can be computed with interval linear algebra.

We have hence to consider not only the $c_\gamma, b_{i,\beta}$ as indeterminates, but also the $a_{i,\alpha}$. Hence equations 1 are quadratic equations. There are moreover further constraints given by the Toeplitz-like structure, some of the c_γ are zero, and the initial intervals for the $a_{i,\alpha}$ are further linear constraints.

Assume now that c, that is, as we recall, an interval plus an internal double-float value, has the internal value positive; then we add a further constraint $c \geq 0$, and we try to minimize c.

If $c = 0$ is realizable, we kill the leading coefficient of g (we'll have to keep track of the additional constraints between the $a_{i,\alpha}$ deriving from this added constraint when trying to kill other coefficients), otherwise we find an optimal value for c, i.e. a new lower bound for the coefficient of g, that is proved to be non zero, and the Buchberger algorithm can continue.

To solve the optimization problem involved in this refinement of the interval c one might use general methods of quadratic optimization; we are currently investigating this and other ad-hoc methods.

Another point that can be included in at least some variants of the algorithm, is that we don't need to handle the case of a possibly zero leading coefficient immediately when it appears; it is also possible to delay the decision, hoping that later another polynomial appears with the same leading term but a better leading coefficient, with a confidence interval not containing zero. This resembles a feature of tangent cone algorithm, in which a decision on a simplification is delayed when it would increase the écart, see (Mora et al. , 1991).

The details of these ideas still have to be worked out completely, and experiments and implementations need to be carried out.

7. Conclusions

We have shown two new algorithms generalizing Buchberger algorithm in a way that allows to handle inexact input for unstable systems of equations. Both rely on syzygies, the first generalizes the trace lifting algorithm, has been completely implemented, has proved to be reliable in some well-known test cases, and will be available in the next release of the PoSSoLib; for the second, the implementation has been completed up to the finding of a "may be zero" leading coefficient; but the handling of this coefficient, with a decision on the realizability of the zero or on the finding of better confidence intervals through quadratic optimization has still to be completed.

References

Caboara, M., Traverso, C., Efficient algorithms for ideal operations, ISSAC98, ACM press (1998)

Gianni, P., Seppälä, M., Silhol, R., Trager, B., Riemann surfaces, plane algebraic curves and their period matrices, J. Symb. Comput. 26, No.6, 789-803 (1998)

Granlund, T., The GNU multiprecision package, `http://www.gmp.org` (1991-2002)

Mora, T., Pfister, G., Traverso, C., An introduction to the tangent cone algorithm, Issues in non-linear geometry and robotics, C. Hoffman, ed., JAI Press (1991)

Mourrain, B., FRISCO test suite `http://www.inria.fr/saga/POL/` (1996-2002)

Traverso, C., Gröbner trace algorithms, pp. 125-138, ISSAC 88, LNCS 358, Springer Verlag (1988)

Traverso, C., Groebner bases of of inexactly known instable systems, Workshop on Under- and Over-Determined Systems of Algebraic or Differential Equations (ADE), Karlsruhe (2002)

Traverso, C., Zanoni, A., Numerical stability and stabilization of Groebner basis computation, ISSAC 2002, ACM press, (2002)

Comprehensive Gröbner Bases and Regular Rings

Dedicated to Bruno Buchberger

Volker Weispfenning

Fakultät für Mathematik und Informatik
Universität Passau
D-94030 Passau, Germany
e-mail: weispfen@uni-passau.de

Abstract

Commutative von Neumann regular rings can be viewed as certain subdirect products of fields. So in some sense they can code arbitrary sets of fields. It was shown in 1987 that most of the Gröbner basis theory over fields initiated by B. Buchberger can be extended to finitely generated ideals over commutative von Neumann regular rings. On the other hand the construction of Comprehensive Gröbner Bases (CGBs) over fields shows that the Gröbner basis theory over fields can be extended to polynomials with parametric coefficients. Here we show that there is a surprisingly close relationship between Comprehensive Gröbner bases over fields and non-parametric Gröbner bases over commutative von Neumann regular rings. Thus the latter can be viewed as an alternative to CGBs. Moreover we show that Gröbner bases over commutative von Neumann regular rings do in fact also cover parametric Gröbner bases over these rings. These facts offer also new algorithmic perspectives on parametric Gröbner bases. They form a strong generalization of the earlier results of Y. Sato and A. Suzuki.

KEYWORDS: comprehensive Gröbner bases, von Neumann regular rings, uniformity

1. Introduction

For every concept and construction in computer algebra the question of uniformity in the input parameters is of crucial importance both from a theoretical and practical viewpoint. This applies in particular to the concept of Gröbner bases

and their construction via some variant of the Buchberger algorithm. Since their invention by Bruno Buchberger in 1965 this concept and the associated constructions and applications have turned out to be of central importance in algorithmic commutative algebra and algebraic geometry. Concerning the dependency of Gröbner bases on term orders uniformity has been achieved by the construction of universal Gröbner bases and the related Gröbner fans Weispfenning [1987a], Mora and Robbiano [1988]. Concerning the dependency of Gröbner bases on the coefficients of the input polynomials uniformity has been achieved by the construction of comprehensive Gröbner bases and the associated Gröbner systems Weispfenning [1992]. In its original version comprehensive Gröbner bases apply to finite sets of polynomials, whose coefficients are elements of some polynomial ring $K[U_1, \ldots, U_m]$ in finitely many parameters over a field K and specializations of the parameters U_i. Recently both the concept and the construction have been generalized to finite sets of polynomials with coefficients in an arbitrary domain R and specializations that are arbitrary homomorphisms of R into some field K' Weispfenning [2002]. In this general form comprehensive Gröbner bases can be viewed as coding simultaneously a whole family of Gröbner bases in polynomial rings over the different base fields arising from all specializations of R.

This flavour of uniformity is also present in the concept of Gröbner bases in polynomial rings over commutative von Neumann regular rings. In the following a *regular ring* always means a commutative von Neumann regular ring. Up to isomorphisms these rings are subdirect products of field that are closed under formation of componentwise inverses of elements with the convention that $0^{-1} = 0$. So they also code in some sense all the fields arising as factors in the subdirect product (compare Saracino and Weispfenning [1975], Loullis [1979] for more precise versions of this heuristic principle). A construction of Gröbner bases for ideals in polynomial rings over regular rings R was established in Weispfenning [1987b] and further optimized in Sato [1998]. It is not surprising that these Gröbner bases code in some sense a whole family of Gröbner bases in the polynomial rings over the fields arising as factors in a subdirect product representation of R. So if the family of these factor fields is large enough these Gröbner bases should be able to serve as a replacement for comprehensive Gröbner bases. This observation was recently made for the classical setting of a ring of multivariate polynomials over a parameter ring $R = K[U_1, \ldots, U_m]$, where K is a field in Sato and Suzuki [2002]. They embed R in a natural way into the regular ring $\overline{K}^{\overline{K}^m}$, where $\overline{K}$ is the algebraic closure of K, by passing from polynomials to polynomial functions, and then construct in an explicit fashion the regular closure S of R in the regular ring $\overline{K}^{\overline{K}^m}$. Then they show that a Gröbner basis G in a multivariate polynomial ring over S can serve as a kind of parametric Gröbner basis for an ideal in a multivariate polynomial ring over R.

Here we investigate the connections between comprehensive Gröbner bases and Gröbner systems one the one hand side and Gröbner bases over regular rings in much greater generality. We describe a number of algorithmic transitions

between these concepts that show a surprisingly close relationship between both under very general conditions.

2. Von Neumann Regular Rings and $*$-rings

In this section we review some facts that are presented in detail in Saracino and Weispfenning [1975]. All rings in this paper will be commutative rings with 1. A ring R is *(von Neumann) regular* if for every $a \in R$ there exists $b \in R$ with $a^2b = a$; $a^* = a \cdot b$ and $a^{-1} = a \cdot b^2$ are then uniquely determined by a and satisfy $a \cdot a^* = a$, $a \cdot a^{-1} = a^* \cdot a^*$ is *the idempotent of* a, a^{-1} *the quasi inverse of* a. $B(R)$ denotes *the Boolean algebra of idempotents of* R (with the operations defined by $\sim a = 1 - a$, $a \sqcap b = a \cdot b$, $a \sqcup b = a + b - a \cdot b$). Any regular ring is a $*$-*ring,* i.e. a ring with an operation $a \longmapsto a^*$ associating with a the smallest idempotent $e = a^* \in B(R)$ with $e \cdot a = a$. Examples of regular rings ($*$-rings) are direct products R of fields (of integral domains), where $a^{-1}(a^*)$ is the pointwise inverse of a [with $0^{-1} = 0$] (the characteristic function of a). More generally, any subring of R closed under $^{-1}$ (under *) is a regular ring (a $*$-ring). Conversely, any regular ring (any $*$-ring) can be canonically represented in this way as subdirect product of fields (of integral domains):

Let for a ring R $Spec(R)$ denote the *prime spectrum* of R i. e. the set of all prime ideals of R, and let $Spec(B(R))$ be the set of all boolean prime ideals of $B(R)$. In a $*$-ring R a $*$-*ideal* of R is an ideal of R that is closed under the operation $*$. We denote the set of prime $*$-ideals of R by $Spec^*(R)$. In a regular ring R every ideal is a $*$-ideal, and so $Spec(R) = Spec^*(R)$. Let now R be an arbitrary $*$-ring. Then the map $Spec^*(R) \longrightarrow Spec(B(R))$, $p \mapsto p^* := \{a^* \mid a \in p\} = p \cap B(R)$ is a bijection. For given boolean prime ideal q the unique preimage under this map is $q^\sim := \{a \in R \mid a^* \in q\}$. As a consequence $Spec^*(R)$ can be identified with Boolean space $Spec(B(R))$ of prime ideals of the Boolean algebra $B(R)$. Then the Stone representation of $B(R)$ extends uniquely to representation of R as subdirect product of factors $R_p := R/p^\sim$, where $p \in Spec(B(R))$. In this representation, the support of any element of R is a clopen set equal to the support of a^* and of a^{-1}. For a subset Q of R and $p \in Spec(B(R))$ we let $\kappa_p : R \longrightarrow R_p$ be the canonocal homomorphism, and Q_p the image of Q under κ_p.

For the present paper, our interest in $*$- rings arises from the following fact: Let R be a $*$-ring and let $S = R[X_1, \ldots, X_r]$ be a polynomial ring over R. Then S is a $*$-ring with $B(S) = B(R)$ and for $f \in S$, f^* is the union of all a^*, where a ranges over the coefficients of f. Morover for all $p \in Spec(B(S)) = Spec(B(R))$, S_p is canonically isomorphic to $R_p[X_1, \ldots, X_r]$ and hence will be identified with this polynomial ring. More generally any homomorphism $\sigma : R \longrightarrow R'$ extends canonically to a homomorphism $\sigma : S \longrightarrow R'[X_1, \ldots, X_r]$ by applying σ coefficientwise.

Similarly, any module M over a $*$-ring R has a canonical representation as

subdirect product of R_p-modules $M_p = p^{\sim} \cdot M$, where $p \in Spec(B(R))$; this applies in particular to ideals I of R.

To conclude, we indicate how (countable) regular ground rings R can be handled computationally: If R is a finite direct product of computable fields, no problem arises. In other cases, R may frequently be regarded as a regular subring of the bounded Boolean power $K[B]$ of a computable field K by the universal countable Boolean algebra B. Then the elements of R are sums $e_1k_1 + \ldots + e_nk_n$ with e_i in B, k_i in K. In the canonical representation, the elements of R are locally constant functions from Cantor space C into K. The elements of B can be represented as disjoint unions of basic clopen subsets of $C = 2^N$ coded by finite strings of zeros and ones.

3. Gröbner Bases over Regular Rings

For the Gröbner basis theory over regular rings we extend the notation of Becker et al. [1998] for the Gröbner basis theory over fields:

Let R be a regular ring, let $S = R[X_1, \ldots, X_r]$ be a polynomial ring over R; for any $p \in SpecB(R)$, we let $S_p = R_p[X_1, \ldots, X_r]$ be the canonical factor of the $*$- ring S at p. T denotes the set of terms $s, t, t', \ldots$, i.e. of power-products of the indeterminates X_i. Monomials are products $a \cdot t$ with $0 \neq a \in R$, $t \in T$. A monomial $a \cdot t$ occurs in a polynomial $f \in S$ if $a \cdot t$ is a summand in f; t occurs in f for some $0 \neq b \in R$. For fixed term order $<$ on T we let $HT(f)$, $HM(f)$, $HC(f)$, $HI(f)$ denote the highest term occuring in f, the highest monomial occuring in f, the highest coefficient a of f (i.e. the coefficient of $HT(f)$), and the highest idempotent a^* of f, respectively. f is quasimonic if $HC(f)$ is an idempotent.

The most important fact about term orders and their induced quasiorders is that both are Noetherian, i.e. do not admit infinite decreasing chains of elements. This is a well-known consequence of Dickson's lemma (compare Becker et al. [1998]. Reduction relations on S (with respect ot a fixed term order $<$ on T) can be defined verbatim as for polynomials over fields, with inverses replaced by quasiinverses.

As in the case of polynomials over fields we have:

THEOREM 3.1: *For any finite list F of non-zero polynomials in S, the reduction $\underset{F}{\longrightarrow}$ is Noetherian.*

COROLLARY 3.1: *Let F be a finite list of polynomials in S: Then the following assertions about the reduction relation $\longrightarrow$ mod F are equivalent:*

(1) $\longrightarrow$ is confluent.

(2) $\longrightarrow$ is locally confluent.

(3) $\longrightarrow$ *has the Church-Rosser property.*

(4) Every $g \in S$ *has a unique normal form mod* F.

In order to avoid certain problems that do not occur for polynomials over fields we restrict our attention from now on to Boolean closed sets of polynomials. The corresponding definitions are as follows: A polynomial $q \in S$ is *Boolean closed* (b.c.) if $q = HI(q) \cdot q$. So for any $q \in S$, $HI(q) \cdot q$ is b.c.. We call $HI(q) \cdot q$ the *Boolean closure* $BC(q)$ of q, and $(1-HI(q)) \cdot q$ the *Boolean remainder* $BR(q)$ of q. So for $q \neq 0$, $BR(q) < q$ and $q = BC(q) + BR(q)$. A finite list Q of polynomials in S is *Boolean closed (b.c.)* if every member q of Q is b.c. An easy algorithm (see Saracino and Weispfenning [1975]) produces from a given finite set $Q \in S$ a finite Boolean closed set $BC(Q)$ in S such that Q and $BC(Q)$ generate the same ideal in S; moreover for every $p \in Spec(B(R))$ we have $Q_p = (BC(Q))_p$. We call $BC(Q)$ the *Boolean closure* of Q.

Among the many equivalent definitions of a Gröbner basis for polynomial ideals over fields, the following is quite natural: A finite list G of polynomials is a Gröbner basis (GB), if for every $f \in (G)$, $f \xrightarrow[G]{*} 0$. Wo we take this as definition of a Gröbner basis in S as well. For finite b.c. sets G, the equivalent definitions known for polynomials over fields are still valid.

LEMMA 3.1: *Let* G *be a finite b.c. set of non-zero polynomials in* S*, and let* $\longrightarrow$ *be the induced reduction. Then the following assertions are equivalent:*

(1) G *is a Gröbner basis.*

(2) For all $0 \neq f \in (G)$, f *is reducible mod* G.

(3) For all $f, g \in S$, $f - g \in (G)$ *implies* $f \downarrow g$.

(4) $\longrightarrow$ *is confluent.*

The following characterization (see Saracino and Weispfenning [1975]) of Gröbner bases G in terms of their images G_p will be of great importance in the following. It uses essentially the compactness of the Boolean space $Spec(B(R))$.

THEOREM 3.2: *Let* G *be a finite b.c. set of non-zero polynomials in* S*. Then* G *is a Gröbner basis iff for all* $p \in Spec(B(R))$, G_p *is Gröbner basis in* S_p.

4. Comprehensive Gröbner Bases

We begin with a very general definition of comprehensive Gröbner bases.

Definition: Let R be a ring, let $R' = R[X_1, \ldots, X_n]$, let $<$ be a term order on the set T of terms in R', let Σ be a class of ring, and let G be a finite subset of R'. Then G is a *comprehensive Gröbner basis* wrt. $<$ and Σ if for all homomorphisms $\delta : R \longrightarrow S$ with $S \in \Sigma$, $\delta(G)$ is a Gröbner basis in $S' = S[X_1, \ldots, X_n]$. Here $\delta : R' \longrightarrow S'$ is the coefficientwise extension of δ and Gröbner bases in S' are

defined as strong Gröbner bases.
If in addition I is an ideal in R' and G is a comprehensive Gröbner basis in R' wrt. $<, \Sigma$, then we say G is a comprehensive Gröbner basis of I wrt. $<, \Sigma$, if I is the ideal generated by G in R'. When Σ is the class of fields we may also omit the reference to Σ.

In almost all cases considered so far in the literature, Σ is the class of fields (except in Sato and Suzuki [2001], where Σ is the class of regular rings). In the by now classical case Weispfenning [1992], R is moreover a polynomial ring $K[U_1, \ldots, U_m]$ over a field or domain K. In Weispfenning [2002] this was generalized to an arbitrary domain R; this paper also provides a construction of a comprehensive Gröbner basis from a given finite ideal basis in R'.
On the negative side it was shown in Weispfenning [1992] that for $R = \mathbb{Z}$ an arbitrary term order $<$ and $\Sigma = \{\mathbb{Z}\}$ there are ideals in R' that have no comprehensive Gröbner basis wrt. $<$ and Σ.

Here we note the following positive results:

THEOREM 4.1: *1. Let R be a field, $R' = R[X_1, \ldots, X_n]$, $<$ a term order on the set T of terms in R and let G be a Gröbner basis in R' wrt. $<$. Then G is also a comprehensive Gröbner basis wrt. $<$.*

2. Let R be a ring, $R' = R[X_1, \ldots, X_n]$, $<$ a term order on the set T of terms in R and let G be a comprehensive Gröbner basis in R' wrt. $<$, and let $H := \{ag \mid g \in G, a \text{ a coefficient of } g\}$. Then H is a comprehensive Gröbner basis wrt. $<$ and the class Σ' of regular rings.

Proof 1.) Since every homomorphism σ of R into a field K is an embeddding, $\sigma(G)$ is also a Gröbner basis in $K[X_1, \ldots, X_n]$, since the Gröbner basis property is preserved by ground field extensions (see Becker et al. [1998]).
2. Let S be a regular ring, $S' = S[X_1, \ldots, X_n]$, let $\varphi : S \longrightarrow \prod_{p \in Spec(S)} S/p$ be the canonical representation of S a subdirect product of fields S/p, and let for $q \in Spec(S)$, $\pi_q : \prod_{p \in Spec(S)} S/p \longrightarrow S_q$ be the projection on the q-th factor. Then for any homomorphism $\delta : R \longrightarrow S$ and every $p \in Spec(S)$, $\pi_p \circ \varphi \circ \delta : R \longrightarrow S/p$ is a specialization of R into a field S/p and so by our hypotheses $\pi_p \circ \varphi \circ \delta(G)$ is a Gröbner basis wrt. $<$ in $S/p[X_1, \ldots, X_n]$. The same applies to H; moreover $\delta(H)$ is "essentially" boolean closed. Consequently by Weispfenning [1987b], $\delta(H)$ is a Gröbner basis wrt. $<$ in $S' = S[X_1, \ldots, X_n]$. This proves the theorem. □

A comprehensive Gröbner basis of an ideal I in in $R' = R[X_1, \ldots, X_n]$ with respect to a term order $<$ and a class Σ of rings is as a rule constructed via a *Gröbner system*. A fairly general definition of such systems is as follows: Let GS be a finite set of triples (P, Q, F), where P and Q are finite subsets of R, and F is a finite subset of R'. Then GS is a Gröbner system for I wrt. $<$ and Σ if for every homomorphism $\sigma : R \longrightarrow S$ with $S \in \Sigma$ there exists at least one triple $(P, Q, F) \in GS$, such that for all $p \in P,\ \sigma(p) = 0$, for all $q \in Q,\ \sigma(q) \neq 0$, and

$\sigma(F)$ is a Gröbner basis of the ideal generated by $\sigma(I)$ in $S' = S[X_1, \ldots, X_n]$. If moreover for all triples $(P, Q, F) \in GS$, $F \subseteq I$, then we call GS a *faithful* Gröbner system for I wrt. $<$ and Σ. From every faithful Gröbner system for I wrt. $<$ and Σ a comprehensive Gröbner basis G for I wrt. $<$ and Σ is simply obtained by putting $G = \{F \mid (P, Q, F) \in GS\}$. For non-faithful Gröbner systems this is not generally true.

5. Connections

In this section we explore the connections between comprehensive Gröbner bases and Gröbner systems on the one hand and Gröbner bases over regular rings on the other hand.

We begin with the passage from a comprehensive Gröbner basis to a Gröbner basis over a regular ring:

THEOREM 5.1: *Let R be a domain, $R' = R[X_1, \ldots, X_n]$, T the set of terms in R' and $<$ a term order on T. Let S be an arbitrary regular ring extending R, let G be a comprehensive Gröbner basis in R' wrt. $<$, and let H be the boolean closure of G in $S' = S[X_1, \ldots, X_n]$. Then H is a Gröbner basis wrt. $<$ in $S' = S[X_1, \ldots, X_n]$.*

Proof Let $S \hookrightarrow \prod_{p \in Spec(B(S))} S_p$ be a representation of S as subdirect product of fields and let $\kappa_p : S_p$ denote the canonical homomorphisms. Since each $\kappa_p \mid R : R \longrightarrow S_p$ is a specialization of R, it follows that for the natural extension

$$\kappa_p | R : R[X_1, \ldots, X_n] \longrightarrow S_p[X_1, \ldots, X_n]$$

$\kappa_p | R(G)$ is a Gröbner basis in $S_p[X_1, \ldots, X_n]$ for every $p \in Spec(B(S))$. By a remark above we have moreover that for each $p \in Spec(B(S))$ $\kappa_p | R(G) = \kappa_p | R(H)$. Since H is boolean closed, this entails by theorem 3.2, that H is a Gröbner basis in S'. □

Together with the standard passage from a faithful Gröbner system GS to the associated comprehensive Gröbner basis $G := \{F \mid (P, Q, F) \in GS\}$, this theorem provides in addition a passage from a faithful Gröbner system to a Gröbner basis over a regular ring.

The next theorem describes more generally a passage from an arbitrary - not necessarily faithful - Gröbner system to a Gröbner basis over a regular ring:

THEOREM 5.2: *Let R be a domain, $R' = R[X_1, \ldots, X_n]$, T the set of terms in R', $<$ a term order on T, and let I be an ideal in R'. Let S be an arbitrary regular ring extending R, let GS be a Gröbner system for I in R' wrt. $<$, and let G be obtained from GS as follows:*

$$G = \bigcup_{(P,Q,F) \in GS} \{\prod_{h \in P}(1 - h^*) \prod_{q \in Q} q^* \; f \mid f \in F\}$$

Then the Boolean closure H of G in $S' = S[X_1, \ldots, X_n]$ is a Gröbner basis of the extension ideal J of I wrt. $<$ in $S' = S[X_1, \ldots, X_n]$.

Proof Let again $S \hookrightarrow \prod_{p \in Spec(B(S))} S_p$ be a representation of S as subdirect product of fields and let $\kappa_p : S_p$ denote the canonical homomorphisms. Let again

$$\kappa_p | R : R[X_1, \ldots, X_n] \longrightarrow S_p[X_1, \ldots, X_n]$$

denote the natural extension of $\kappa_p | R$. Call a triple $(P, Q, F) \in GS$ *good* wrt. κ_p, if it has the property that for all $h \in P,\ \kappa_p(h) = 0$, for all $q \in Q,\ \kappa_p(q) \neq 0$, and $\kappa_p(F)$ is a Gröbner basis of the ideal generated by $\kappa_p(I)$ in $S_p[X_1, \ldots, X_n]$ wrt. $<$. Notice that for all non-good triples $(P', Q', F') \in GS$ and all $f \in F$ we have $\kappa_p(\prod_{h \in P}(1 - h^*) \prod_{q \in Q} q^*\ f = 0$. Consequently

$$\kappa_p(G) = \bigcup \{\kappa_p(F) \mid (P, Q, F) \in GS \text{ is good}\}$$

is a Gröbner basis of the ideal generated by $\kappa_p(I)$ in $S_p[X_1, \ldots, X_n]$. ¿From this fact we conclude as in the previous proof that H is a Gröbner basis in S'. □

The next theorem shows that conversely Boolean closed Gröbner bases over regular rings are in fact automatically comprehensive Gröbner bases:

THEOREM 5.3: *Let S be a regular ring, let $S' = S[X_1, \ldots, X_n]$, let T the set of terms in S' and $<$ a term order on T. Let G be a Boolean closed Gröbner basis in S' wrt. $<$. Then G is also a comprehensive Gröbner basis wrt. $<$ and the class Σ' of regular rings.*

Proof By theorem 4.1, 2. it suffices to show that G is also a comprehensive Gröbner basis wrt. $<$ and the class Σ of fields. Let K be a field and let $\sigma : S \longrightarrow K$ be a homomorphism. Then the kernel of σ is a prime ideal in S, and hence of the form p for some $p \in Spec(B(R))$. Thus the factor ring S_p embeds into K. Modulo an isomorphism we may assume that in fact $\sigma(S) = S_p \subseteq K$. By theorem 3.2 the image $\sigma(G)$ is a Gröbner basis in $S_p[[X_1, \ldots, X_n]$ wrt. $<$, and hence also in $K[[X_1, \ldots, X_n]$ wrt. $<$ by the lemma on ground fields extensions. □

This fact can be generalized to arbitrary rings R without nilpotent elements as ground rings, in particular to domains R :

COROLLARY 5.1: *Let R be a ring without nilpotent elements. We regard R as subring of $\prod_{p \in Spec(R)} Quot(R/p)$ via the embedding φ mapping $r \in R$ to $\{r + p \mid p \in Spec(R)\}$. Let S be the regular closure of R in the regular ring $\prod_{p \in Spec(R)} Quot(R/p)$. Let I be an ideal in $R' = R[X_1, \ldots, X_n]$, let J be the extension ideal of I in $S' = S[X_1, \ldots, X_n]$. Let $<$ be a term order and let G be a boolean closed Gröbner basis of J wrt. $<$ in S'. Then for every specialization $\sigma : R \longrightarrow K,\ \sigma(G)$ is defined and constitutes a Gröbner basis for the ideal generated by I in $K[X_1, \ldots, X_n]$.*

Proof By the fact that R has no nilpotent elements, R embeds indeed via φ into $\prod_{p\in Spec(R)} R/p$. So we may regards R as subring of the regular ring $\prod_{p\in Spec(R)} \mathrm{Quot}(R/p)$, and thus form the regular closure S of R in $\prod_{p\in Spec(R)} \mathrm{Quot}(R/p)$. Then it follows from Saracino and Weispfenning [1975] that for all $p \in Spec(R)$, $R/p = S_{p^*}$. Let now K be a field and let $\sigma : R \longrightarrow K$ be a homomorphism. Then the kernel of σ is a prime ideal in R, and hence in $Spec(R)$. Thus the factor ring $R/p = S_{p^*}$ embeds into K. Modulo an isomorphism we may assume as above that in fact $\sigma(S) = S_{p^*} \subseteq K$. By theorem 3.2 the image $\sigma(G)$ is a Gröbner basis in $S_{p^*}[[X_1, \ldots, X_n]$ wrt. $<$, and hence also in $K[[X_1, \ldots, X_n]$ wrt. $<$ by the lemma on ground fields extensions. □

The main result of Sato and Suzuki [2002] is a essentially a special case of this corollary. They take $R = K[U_1, \ldots, U_m]$ as a polynomial ring in parameters U_i over a field K and S as the regular closure of R in the regular ring $\overline{K}^{\overline{K}^m}$, where $\overline{K}$ is the algebraic closure of K. By Hilbert's Nullstellensatz this ring can be taken as a substitute for the ring $\prod_{p\in Spec(R)} \mathrm{Quot}(R/p)$. A central part of the paper consists then in an explicit construction of the regular closure of R in $\overline{K}^{\overline{K}^m}$.

There is still another variant of this result that under the same hypotheses provides a passage from G to a Gröbner sytem GS in R'. A *Boolean specialization* of $B(S)$ is a Boolean homomorphism $\sigma : B(S) \longrightarrow \{0, 1\}$. Boolean specializations are in one-to-one correspondence to Boolean prime ideals of $B(S)$; so we can identify the set of Boolean specializations of $B(S)$ with $Spec(B(S))$.

Corollary 5.2: *Assume the hypotheses of the previous corollary. For every Boolean specialization $\sigma \in Spec(B(S))$ let $\sigma(G)$ be obtained by replacing every idempotent coefficient in a polynomial in G by its image under σ. Then $GS := \bigcup_{\sigma\in Spec(B(S))} \sigma(G)$ is a Gröbner system of J in R'.*

Proof Immediate from the previous proof. □

Notice that in the last corollary we could also replace $Spec(B(S))$ by the set of all maps $\sigma : B(S) \longrightarrow \{0, 1\}$. This set is higly redundant but computationally easier to handle. Notice also that GS is in general not a faithful Gröbner system. For this to be the case one would need to know that each $\sigma(G) \subseteq J$. This is in fact true if the Gröbner basis G is constructed exactly as described in Weispfenning [1987b]. Thus in this case, we do in fact obtain algorithmically a comprehensive Gröbner basis H in R' from G via the passage through the faithful Gröbner system GS.

6. Conclusions

We have shown that two apparently unrelated theories, namely comprehensive Gröbner bases and Gröbner systems on the one hand side, and Gröbner bases over regular rings are in fact two facets of the same general idea. We have shown a number of algorithmic constructions leading from a concept of the first kind to

an essentially equivalent concept of the second kind, and vice versa. Our results form a strong generalization of the results in Sato and Suzuki [2000, 2001, 2002] that connect the two aspects in special situations.

The transitions from one aspect to the other could also provide new ideas on more effient constructions for Gröbner bases for parametric polynomials.

References

Thomas Becker, Volker Weispfenning, and Heinz Kredel. *Gröbner Bases, a Computational Approach to Commutative Algebra*, volume 141 of *Graduate Texts in Mathematics*. Springer, New York, corrected second printing edition, 1998.

George Loullis. Sheaves and boolean valued model theory. *J. Symbolic Logic*, 44 (2):153–183, 1979.

T. Mora and L. Robbiano. Gröbner fan of an ideal. *Journal of Symbolic Computation*, 6:183–208, 1988.

D. Saracino and V. Weispfenning. On algebraic curves over commutative regular rings. In *Model Theory and Algebra*, volume 498 of *LNM*, pages 307–383. Springer, 1975.

Y. Sato and A. Suzuki. Gröbner bases in polynomial rings over von Neumann regular rings - their applications. In *ASCM'2000, The 4th Asian Symposium on Computer Mathematics*, Lecture Notes Series on Computing, Singapore/River Edge, December 2000. World Scientific Publ.

Y. Sato and A. Suzuki. Discrete comprehesive Gröbner bases. In B. Mourrain, editor, *ISSAC'2001*, pages 292–206, New York, 2001. ACM-Press.

Y. Sato and A. Suzuki. An alternative approach to comprehensive Gröbner bases. In T. Mora, editor, *ISSAC'2002*, New York, 2002. ACM-Press.

Yosuke Sato. A new type of canonical Gröbner bases in polynomial rings over vonNeumann regular rings. In O. Gloor, editor, *ISSAC'98*. ACM-Press, August 1998.

V. Weispfenning. Constructing universal Gröbner bases. In *Proceedings AAEEC, Menorca*, volume 356 of *LNCS*. Springer Verlag, 1987a.

Volker Weispfenning. Gröbner bases for polynomial ideals over commutative regular rings. In *Proceedings: EUROCAL'87 (Leipzig 1987)*, volume 378 of *Lecture Notes of Computer Science*, pages 336–347. Springer, 1987b.

Volker Weispfenning. Comprehensive Gröbner bases. *Journal of Symbolic Computation*, 14:1–29, July 1992.

Volker Weispfenning. Canonical comprehensive Gröbner bases. In T. Mora, editor, *ISSAC 2002*. ACM Press, 2002.

An Automated Prover for Zermelo-Fraenkel Set Theory in *Theorema*

Wolfgang Windsteiger*

RISC Institute
A-4232 Hagenberg, Austria
E-mail: Wolfgang.Windsteiger@RISC.Uni-Linz.ac.at

Abstract

This paper presents some fundamental aspects of the design and the implementation of an automated prover for Zermelo-Fraenkel set theory within the well-known *Theorema* system. The method applies the "Prove-Compute-Solve"-paradigm as its major strategy for generating proofs in a natural style for statements involving constructs from set theory.

KEYWORDS: Automated Theorem Proving, Set Theory, Theorema

1. Introduction

The set theory prover in *Theorema* adapts the "**P**rove-**C**ompute-**S**olve" (short: PCS) prove strategy for proofs containing language constructs from set theory. The PCS paradigm was introduced originally in (3) and it has already been applied successfully for proofs in elementary analysis in (12). The main strategy in a PCS-oriented prover is to structure the proof generation into phases of

- proving (P), i.e. application of logical inference rules for propositional connectives and for quantifiers,
- computing (C), i.e. rewriting w.r.t. formulae in the knowledge base,
- solving (S), i.e. instantiation of existential variables.

Having the computer algebra system Mathematica in the background of *Theorema*, we aim towards applying known solution methods from computer algebra during the S-phase, such as the Gröbner bases method for systems of algebraic equations or Collins' CAD method for systems of inequalities over the reals.

*This work has been supported by the "SFB Numerical and Symbolic Scientific Computing" (F013) at the University of Linz and the european union "CALCULEMUS Project" (HPRN–CT–2000–00102).

The current design of provers in the *Theorema* system requires a so-called "user prover" to be composed from "special provers" (see (11)). A special prover consists of a collection of inference rules, whereas the user prover guides *the strategy*, through which the proof search procedure applies the inference rules. Consequently, the set theory user prover consists of a *set theory proving unit* handling set-theory-related connectives and quantifiers in the goal or in the knowledge base, a *set theory computing unit*, and a *set theory solving unit*. In addition to these set theory specific components, the set theory prover utilizes several special provers already available in the *Theorema* system, such as `BasicND` for handling basic quantifiers from predicate logic using natural deduction, `QR` for rewriting using quantified formulae in the knowledge base, and `CDP` for applying case distinction (invented in (4); see (12) for detailed description).

Following the philosophy of most of the *Theorema* provers, the set theory prover aims at generating automated proofs in a human-like natural style. Since many mathematicians are used to building up their theories in the frame of set theory, computer support for doing proofs in this area of mathematics is a basic ingredient for computerized mathematics. In our experience, the acceptance of machine-generated proofs depends heavily on the readability of the proof for a human. In the automated theorem proving community, however, this aspect has not played a central role for a long time. Of course, as long as one does not display the proof, one can expand set-theoretic language constructs into first-order predicate logic and then apply powerful first-order theorem provers, like Otter, Vampire, or SPASS. The *Theorema* set theory prover, on the other hand, implements proof strategies applied by humans in an attempt to generate machine-proofs in a style acceptable by a human. Apart from others, this will have enourmous impact on computer-aided maths education.

The description is structured as follows: Section 2 describes the theoretical basis upon which the set theory user prover `SetTheoryPCSProver` is built, Section 3 introduces the set theory proving units `STP` and `STKBR`, Section 4 describes the set theory computing unit `STC`, Section 5 presents the set theory solving unit `STS`, and finally we conclude with some examples of proofs generated by the `SetTheoryPCSProver` in Section 6.

The relation of this work to Bruno Buchberger's work is more than obvious since Bruno Buchberger is the founder and leader of the *Theorema* project, he implemented early prototypes of several provers and the entire *Theorema* system design and development is based on his experience of more than thirty years of doing proofs and teaching students how to do proofs. The set theory specific components of the prover have been developed exclusively by the author in the frame of (13). More details on implementation and a number of case studies using the set theory prover can be found in (13). These units have been embedded into the *Theorema* system through existing mechanisms developed over the years by various members of the *Theorema* working group. Significant contributions from the author went into language design and the design and implementation of the

Theorema rewrite engine, which is applied also during the C-phase in the set theory prover.

2. The Theoretical Basis for the Set Theory Prover

The use of set theory in *Theorema* is not tied to one particular axiomatization of set theory. Instead, we introduce "sets" on the level of the language by providing the braces '{' and '}' as a flexible arity matchfix function symbol used for constructing finite sets and the set quantifier. Providing these language constructs, we implicitly assume that sets such as $\{a\}$, $\{1, b\}$, $\{x \underset{x}{\mid} P_x\}$ (the set of *all* x satisfying P_x), or $\{T_x \underset{x}{\mid} P_x\}$ (the set of *all* T_x, when x satisfies P_x) actually exist, which is typically guaranteed by some axioms of the underlying set theory. There are different approaches, in the Zermelo-Fraenkel axiomatization (ZF) as described e.g. in (5) the existence of the singleton $\{a\}$ follows from an axiom on power sets and the existence of $\{1, b\}$ follows from the existence of singletons together with an axiom on unions, whereas in an axiomatization given in (10), which also follows the spirit of ZF, the existence of $\{1, b\}$ is guaranteed by an axiom of pairing and the singleton $\{a\}$ is then just defined to denote the pair $\{a, a\}$.

A *Theorema* language construct that deserves closer inspection in this context is the so-called *set quantifier*, i.e. the expression $\{x \underset{x}{\mid} P_x\}$, which allows one to define a set from a property P_x ? In the literature, this is often addressed as *the abstraction* of a set from a property and it goes back to G. Cantor, the founder of modern set theory. As explained in (nearly) every introductory course in mathematics, the unrestricted use of abstraction soon leads to contradictions such as the well-known Russel paradox. With R denoting the "Russel-set" $\{x \underset{x}{\mid} x \notin x\}$ it is straight-forward to derive the contradiction $R \in R \Leftrightarrow R \notin R$. ZF set theory resolves this paradox by imposing a certain structure on the formula P_x in an abstraction $\{x \underset{x}{\mid} P_x\}$, which disallows constructions like R. Von-Neumann-Gödel-Bernays' axiomatization (NGB) of set theory (see e.g. (1) or (8)) distinguishes between sets and classes and allows the membership predicate only for sets. Russel's paradox is avoided by showing that R is not a set an therefore $R \in R$ is not a well-formed assertion. Russel himself introduced types as a way out by allowing membership only for sets of different type (see (9)). $R \in R$ is not allowed on the grounds that R and R are not of different type.

The *Theorema* system as such does not force the user into one of the above mentioned axiomatizations. The *Theorema language* allows unrestricted use of both the set quantifier and the membership predicate, therefore allowing both the definition of R and formulae such as $R \in R \Leftrightarrow R \notin R$. The set theory *prover*, however, relies on ZF and therefore refuses to apply inference rules on formulae involving constructs such as R. In other words, the *Theorema* set theory prover

does not support all of what the *Theorema* language offers for set theory. If a user desires to work e.g. in NGB set theory the *Theorema* language would allow this but our set theory prover would not support it.

Among mathematicians *using set theory*, however, there is a common understanding of the intuition behind constructs from set theory, which is more or less independent of its concrete axiomatization. Following this spirit, we provide *definitions of the basic constructs of set theory* supported by the *Theorema* set theory prover, which follow the ZF-style of axiomatizing set theory. This should mean that the consistency of these definitions is guaranteed by axioms of ZF. Thus, the *Theorema* set theory prover should be a useful tool for mathematicians embedding their work in some set theory, which is consistent with these definitions. We do not invent a new set theory that promises to be better suited for automated theorem proving, an approach that is taken elsewhere, e.g. in (6).

The set theory prover is based on the following definitions[†].

Definition:

$$a \in \{x \mid_{x \in S} P_x\} :\Longleftrightarrow a \in S \wedge P_{x \to a} \tag{1}$$

$$a \in \{T_x \mid_{x \in S} P_x\} :\Longleftrightarrow \underset{x \in S}{\exists} (P_x \wedge a = T_x) \tag{2}$$

$$\emptyset := \{x \mid_{x \in S} x \neq x\} \quad \text{(for some set } S\text{)} \tag{3}$$

$$a \in \{a_1, \ldots, a_n\} :\Longleftrightarrow a = a_1 \vee \ldots \vee a = a_n \quad (\text{for } n \geq 1) \tag{4}$$

$$a \in S_1 \cup \ldots \cup S_n :\Longleftrightarrow a \in S_1 \vee \ldots \vee a \in S_n \quad (\text{for } n \geq 2) \tag{5}$$

$$a \in \bigcup S :\Longleftrightarrow \underset{s \in S}{\exists} a \in s \tag{6}$$

$$\bigcup_{\substack{x \in I \\ C_x}} S_x := \bigcup \{S_x \mid_{x \in I} C_x\} \tag{7}$$

$$a \in S_1 \cap \ldots \cap S_n :\Longleftrightarrow a \in S_1 \wedge \ldots \wedge a \in S_n \quad (\text{for } n \geq 2) \tag{8}$$

$$a \in \bigcap S :\Longleftrightarrow \underset{s \in S}{\forall} a \in s \tag{9}$$

$$\bigcap_{\substack{x \in I \\ C_x}} S_x := \bigcap \{S_x \mid_{x \in I} C_x\} \tag{10}$$

$$S_1 \subseteq S_2 :\Longleftrightarrow \underset{a}{\forall} a \in S_1 \Rightarrow a \in S_2 \tag{11}$$

$$S_1 = S_2 :\Longleftrightarrow \underset{a}{\forall} a \in S_1 \Leftrightarrow a \in S_2 \tag{12}$$

We list only the most important definitions. In the concrete implementation, the prover can handle some more like e.g. cross product, power-set, or set difference, see (13). When using the *Theorema* set theory prover one accepts these definitions and assumes an underlying axiomatic system such as ZF that guarantees the existence of all sets defined above.

[†] $P_{x \to a}$ stands for P with each free occurrence of x substituted by a.

2.1. Preliminaries on Terminology

We will use the following terminology in the description of the prove modules: a proof situation $\kappa \vdash G$ is made up from a knowledge base of assumptions κ and a goal G, and it should be understood as an abbreviation for the phrase: "We have to prove G from κ". Typically, the goal will be a single formula of the *Theorema* language, whereas the knowledge base consists of *a collection of formulae*, called the assumptions.

Now, the task of the special provers is essentially the execution of individual *proof steps* that *reduce* the proof situation, where the rules applied by the special provers guiding the transformations of proof situations are called *inference rules*. Thus, an inference rule turns a proof situation $\kappa \vdash G$ into a proof situation $\kappa' \vdash G'$ with a new goal G' and a new knowledge base κ'. In the description of inference rules, we will denote an inference rule named I transforming $\kappa \vdash G$ into $\kappa' \vdash G'$ by

$$I : \frac{\kappa' \ \vdash \ G'}{\kappa \ \vdash \ G}$$

(read as: "The rule I justifies a proof step to reduce the proof of G from κ to a proof of G' from κ'"). This notation is similar to notations used in logic for describing inference rules in formal prove calculi (e.g. the natural deduction calculus or the Gentzen calculus). Certain similarities to these formalisms are desired, but we use it purely as a symbolic description for proof steps, and we do not refer to any meaning of the symbols in any known logic system.

An example of a well-known inference rule written in this style is

$$\textbf{ArbitraryButFixed} : \frac{\kappa \ \vdash \ P_{x \to x_0}}{\kappa \ \vdash \ \underset{x}{\forall} P_x} \quad \text{(where } x_0 \text{ is a new constant)}$$

meaning that, in order to prove $\underset{x}{\forall} P_x$ (from κ) it suffices to prove $P_{x \to x_0}$ (from κ) for a new constant x_0.

3. STP and STKBR: The Set Theory Proving Units

The PCS proof strategy imposes a structure on proofs as alternating phases of proving, computing, and solving, as already described in Sect. 1. *Proving* can in this context be interpreted as *eliminating theory-specific language constructs*, in particular eliminating *quantifiers*. The set theory prover is a prover that can handle language constructs from set theory *in addition* to standard predicate logic. Therefore, it re-uses the special provers available in the *Theorema* system for handling propositional connectives and the $\forall$-quantifier (see (4), (12), and (13)). Set theory specific *proving* is covered by the two new special provers `STP` and `STKBR`. During the Prove-phase, we alternate steps of *reducing the goal* with steps of *expanding the knowledge base*. While `STP` reduces set theory specific

language constructs in the proof-goal, `STKBR` expands them in the knowledge base.

3.1. Inference Rules used in STP

Set theory specific goal reduction is implemented as a special prover named `STP`. As most of the special provers in *Theorema*, `STP` implements individual inference rules as individual function definitions for one overloaded Mathematica function `STP`, which differ in the patterns specifying the parameters describing the proof situation. The choice of which inference rule to apply next, is made by the global *Theorema* proof-search procedure. As its main strategy it applies *pattern matching* on the current proof situation against proof situation patterns defined in one of the special provers. A few inference rules are influenced in addition by global variables used by `STP`, some strategies depend on the proof progress stored in `STP`'s *local proof context*, which is the third parameter in a call to `STP` in addition to the proof-goal and the knowledge base.

The inference rules are grouped into rules for *membership*, rules for *inclusion*, and rules for *set equality*. The rules for membership contain at least one inference rule for each "kind of set" introduced in Def. 2.1, in some cases we provide tailored rules in order to offer special treatment for special cases. We show some of the membership rules as they are used in `STP`.

$$\textbf{MembershipAbstraction}: \frac{\kappa \ \vdash \ t \in s \wedge P_{x \to t}}{\kappa \ \vdash \ t \in \{x \underset{x \in s}{|} P_x\}}$$

We give an impression of what the result of this inference rule is in a concrete example. If, during a concrete proof, the proof search procedure arrives at a proof situation, where we need to prove $a \in \{x \underset{x \in s}{|} x < 10\}$ w.r.t. some knowledge base KB, then the special prover `STP` would be called in the following format:

$$\text{STP}[\bullet\text{lf}[\texttt{"1"}, a \in \{x \underset{x \in s}{|} x < 10\}, \bullet\text{finfo}[]], \bullet\text{asml}[\text{KB}], \text{af}]$$

where $\bullet$lf[...] represents the proof-goal labelled "1", $\bullet$asml[KB] is the current knowledge base, and "af" are the additional facts containing among others `STP`'s local proof context. Note, that this is *not* how the *user* needs to call the prover, the actual call of the special prover is based on internal data structures, which are built-up automatically during the proof search. It is the task of the *Theorema* User Language (see (13)) to serve as an interface between the user and the internal data structures as they show up above. The result of this call is the *new proof situation*

```
{"AndNode",
 {"MembershipAbstraction", .usedFormulae["1"],
   .generatedFormulae[.lf["1'",And[Element[a,s],a<10],.finfo[]]]},
```

```
{{"ProofSituation", .lf["1'",And[Element[a,s],a<10],.finfo[]],
  .asml[kb], af}}, {}, {}, "pending"}
```

The proof search procedure will insert this node into the *Theorema* proof object. The node contains enough information in order to later *simplify* a successful proof (object) and to generate the natural language text from it. Note, however, that it *does not contain* the natural language text representation itself! When later generating the proof presentation from a proof object, this step of the proof would read as follows:
In order to prove (1) we have to show:
(1') $a \in s \wedge a < 10$.

The correctness of the inference rule "MembershipAbstraction" follows immediately from the definition of set abstraction. Some of the inference rules, however, condense several inference steps into one compact rule to be applied. In these cases, we provide hand-proofs for the correctness of the respective rules[‡]. An example of such a rule is the elimination of the union-quantifier in the goal.

$$\textbf{MembershipUnionOf}: \frac{\kappa \vdash \underset{x\in s}{\exists}\ (t \in S_x \wedge C_x)}{\kappa \vdash t \in \underset{\substack{x\in s\\ C_x}}{\bigcup} S_x}$$

MembershipUnionOf reduces the proof of $t \in \underset{\substack{x\in s\\ C_x}}{\bigcup} S_x$ to prove $\underset{x\in s}{\exists}\ (t \in S_x \wedge C_x)$.

Proof: Assume $\underset{x\in s}{\exists}\ (t \in S_x \wedge C_x)$, thus $t \in S_{x_0} \wedge C_{x_0}$ for some constant $x_0 \in s$. With $z := S_{x_0}$ we can infer from this $t \in z \wedge C_{x_0} \wedge z = S_{x_0}$, hence

$$\underset{z}{\exists}\ (\ \underset{x\in s}{\exists}\ t \in z \wedge C_x \wedge z = S_x)\ . \tag{13}$$

Separating the quantifiers in (13) gives $\underset{z}{\exists}\ (t \in z \wedge \underset{x\in s}{\exists}\ (C_x \wedge z = S_x))$, which, by (2), is equivalent to $\underset{z}{\exists}\ (t \in z \wedge z \in \{S_x \mid_{x\in s} C_x\})$. By (6) this is equivalent to $t \in \bigcup\{S_x \mid_{x\in s} C_x\}$, thus $t \in \underset{\substack{x\in s\\ C_x}}{\bigcup} S_x$ by (7). □

Set inclusion reduces, by definition, to membership and set equality reduces to membership. In addition to these reductions, we implemented several inference rules for special cases that reduce the search depth for the proof search, e.g.

$$\textbf{ConjunctionSubset}: \frac{\text{proved}}{\kappa \vdash \{x \mid_{x} \ldots \wedge x \in S \wedge \ldots\} \subseteq S}$$

[‡]Ideally, the *Theorema* Predicate Logic Prover should be capable of producing these proofs when having Def. 2.1 in its knowledge base. Unfortunately, however, some of the definitions, notably (1), and some inference rules "live" on the language expression level and they refer to variable substitution, free variables and the like. In its current status, the *Theorema* language cannot express these things on the object level!

$$\textbf{EqualsEmptySet}: \frac{\kappa \vdash \neg P_{x \to x_0}}{\kappa \vdash \{T_x \mid \underset{x}{} P_x\} = \emptyset} \quad \text{where } x_0 \text{ is some new constant,}$$

and some extensions so that the prover can also deal with cardinality and function properties such as bijectivity. For details see (13).

3.2. The Structure of STKBR

The special prover `STKBR` (for Set Theory Knowledge Base Rewriting) uses a level saturation technique (see also (7)), to infer *new knowledge* from the knowledge base by unfolding definitions of set theoretic language constructs. It differs drastically from most of the other special provers in the *Theorema* system in that it does not implement inference rules as *separate definitions* for *one* Mathematica function. This "classic" implementation scheme for *Theorema* special provers, which introduces one definition per proof situation, is not suitable for an efficient implementation of a level saturation mechanism, because inferring new formulae one at the time would result in a massive growth of the required search depth for the proof search. The `STKBR` function, instead, is implemented as just *one definition*, which produces *all possible new formulae during only one application*. This has the advantage, that several inference rules can be applied in parallel during one `STKBR`-step instead of adding only one new formula at the time to the knowledge base.

As a consequence, the `STKBR` function does not specify the syntactic pattern of the proof situation in its parameters but it is considered to be applicable to the current proof situation as soon as new formulae occur in the knowledge base compared to `STKBR`'s previous run. This check is done with the help of an entry in the local proof context that stores the labels of all assumptions that have already been treated in the preceding saturation level. In case new formulae have been added to the assumptions, the saturation of the current knowledge level happens in two phases:

- In a first phase, new formulae are, if desired, simplified by computation using built-in semantic knowledge available in the *Theorema* language semantics[§], see also `STC` in Sect. 4. In case this type of simplification is not desired, this phase can be skipped through a user option in the call of the prover.
- In a second phase, new knowledge is *inferred* from the simplified new formulae using inference rules for set theory. These inference rules are again grouped into two groups,
  - *Group One* containing rules for inferring new knowledge from *one* known formula and

[§]Here we see that `STKBR` contributes to both the P- and the C-phase, hence, we should not call it a pure proving unit! For reasons of efficiency we allowed this mixture of P- and C-phase in *one* special prover in the current implementation.

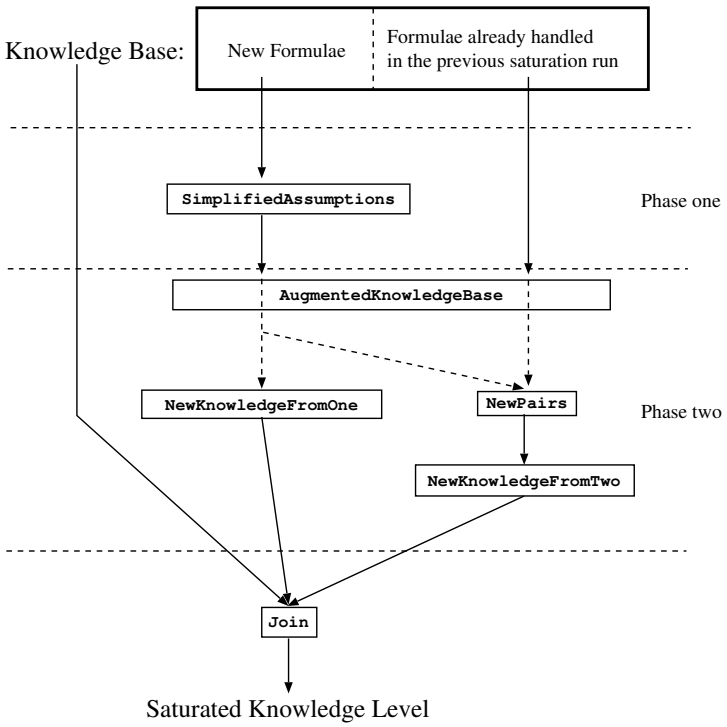


Figure 1: Schematic flow of the STKBR level saturation

– *Group Two* containing rules for inferring new knowledge from *two* known formulae.

Matching rules from Group One are applied to the simplified new formulae, matching rules from Group Two are applied to *all new pairs* of formulae that can be formed using additionally the simplified new formulae[¶].

All formulae generated during these two phases are adjoined to the knowledge base for the new proof situation. The augmented knowledge base is considered to contain *all knowledge*, that can be made available at that point, thus, we call it a *saturated knowledge level*. The schematized flow of STKBR level saturation mechanism is shown in Fig. 1, where the boxed names are the names of the respective functions in the actual implementation.

Phase one is accomplished by calling the function 'SimplifiedAssumptions' with two arguments, the entire knowledge base and a list of labels specifying that part of the knowledge base that has already been used in the previous saturation phase. Each formula from the knowledge base, whose label is *not among the handled labels*, is sent through the function 'EvaluateFromProve',

[¶]Up to now, no inference rules have been implemented that depend on three formulae. As soon as such inference rules are needed, we will provide a Group Three of inference rules, which will be applied to all possible triples of formulae.

which *computes* a simplified version of the formula w.r.t. semantic knowledge from the *Theorema* language. 'EvaluateFromProve' is the function used also in the `STC` module for *goal simplification by computation*, see Sect. 4. Note, that 'EvaluateFromProve' is based on the function 'EvaluateStandard', which is the basic evaluation function for computations using *Theorema* semantics, which is used also by `Compute`, the top-level user function to initiate computations. This guarantees utmost coherence between all computations happening in the *Theorema* system, be it on the user level by calling `Compute`, be it on the prover level by doing simplifications on the goal or on the knowledge base.

Phase two is covered in the implementation by the function 'AugmentedKnowledgeBase', which receives two arguments: the simplified knowledge base resulting from phase one and the list of already handled labels as above. 'NewKnowledgeFromOne' applies Group One of inference rules componentwise to *all new assumptions*, 'NewKnowledgeFromTwo' applies Group Two of inference rules to *all new pairs* that can be formed using the new assumptions and the results are added to the knowledge base. The inference rules applied by `STKBR` can more or less be read off Def. 2.1, hence we do not list them here.

4. STC: The Set Theory Computing Unit

The *Theorema* language contains semantics essentially for *finite sets*, namely

- sets that are constructed using the set braces '{' and '}' as set constructor applied to finitely many arguments, and
- sets that are constructed using *algorithmic versions* of the set quantifier (see also (2)), i.e. set quantifiers with finite and computable range specifications (see (13)). In particular, *integer ranges* and *set ranges* for finite sets are algorithmic ranges, which lead to finite sets when used in combination with the set quantifier.

The *Theorema* semantics enables the *construction* of finite sets as an enumeration of the (finitely many) elements contained in the set. Set operations (such as union, intersection, power set, etc.) on finite sets are implemented in a constructive fashion. *Proving properties* (such as membership, inclusion, or set equality) of finite sets therefore reduces to *testing finitely many cases*, which is implemented in the frame of the *Theorema* language as well.

From the user's point of view, computation using built-in semantics knowledge is available in the *Theorema* system through the top-level user function `Compute`. A typical computation involving finite sets is

$$\text{Compute}[\{3x \mid_{x\in\{1,2,3,4\}} \text{is-prime}[x]\}]$$

resulting in the finite set $\{6, 9\}$.

It is the intention of the `STC` special prover to integrate the knowledge available for computations seamlessly into the *Theorema* proving machinery. Otherwise,

all algorithmic knowledge about finite sets needed to be re-implemented inside the set theory prover, which would make it next to impossible to guarantee identical behavior in proving and computing. In order to avoid this duplication of code and knowledge, the `STC` prover simplifies the goal by sending the formula to the same evaluation function that is also used in `Compute` and in `STKBR`.

Basically, when the `STC` prover applies to a proof situation, one proof step consists of calling the evaluation function 'EvaluateFromProve' (see also Sect. 3.2) and, in case the result differs from the original form, of adding a node to the proof object, from which the effect and a complete trace of the computation can be displayed. Again, the use of 'EvaluateFromProve' preserves coherence with `STKBR` and `Compute`. Many details on combining computation with proving can be found in (13).

5. STS: The Set Theory Solving Unit

The special prover `STS` collects inference rules for eliminating existential quantifiers[‖]. Methods used for instantiating existential goals range from *matching* against formulae in the knowledge base, over *unification* and *introduction of solve constants* until to employing the Mathematica 'Solve' function to obtain solutions of equational goals. We present only one typical inference rules from `STS`.

$$\textbf{IntroSolveConstant}: \frac{\kappa \vdash Q_{y\to y^*} \wedge \underset{x\in s}{\exists} (P_x \wedge y^* = T_x) \wedge R_{y\to y^*}}{\kappa \vdash \underset{y}{\exists} (Q_y \wedge y \in \{T_x \mid x\in s} P_x\} \wedge R_y)}$$

where Q_y and R_y are possibly empty conjunctions of formulae and y^* is a *solve constant*.

A solve constant[**] is some constant, which we still have to assign a concrete value. Solve constants are introduced in order to eliminate existential quantifiers by substituting a constant for the quantified variable, where at the moment of introducing the constant, its concrete value can not yet be determined. For the proof to succeed, *all solve constants* that have been introduced must be eliminated by substituting appropriate *ground terms* in such a manner that the resulting formula can be proven. Of course, the strategy after introducing solve constants must always be to isolate the solve constants, which is typically done by *solving*, using methods depending on the nature of the remaining formula. Applying this strategy reduces proving to solving over various domains, and it

[‖]In fact, it should contain only the set theory specific part of solving. Since the solving components in the *Theorema* system are not yet far-advanced, we started with `STS` collecting inference rules for proof situations as they appear in typical proofs in set theory.

[**]What we call solve constant is often addressed as *meta variable* by other authors. The technique of meta variables is well known and used also in other systems. Essentially, it imitates what a human does when instantiating existential quantifiers, in particular, in the well-known proof on limits, continuity, etc.

offers the possibility to benefit from the great advances that have been accomplished in developing powerful solution methods in computer algebra.

The inference rule described above might appear random. It is part of `STS` since it applies exactly to proof situations left after expanding membership in a union, i.e. goals of the form $t \in \bigcup\{T_x \mid_{x\in s} P_x\}$. The rule eliminates the outermost existential quantifier, but it introduces another existential quantifier. `STS` contains further rules, which allow the elimination of the existential quantifier in this particular and even in other more general situations (see (13)). In addition to rules introducing solve constants, the `STS` prover, of course, also contains several rules for instantiating solve constants as soon as they appear in an isolated position.

6. Two Examples of Automatically Generated Proofs

This section contains representative proofs that were generated completely automatically by the *Theorema* set theory prover. The optical appearance of the proofs in the system corresponds exactly to how they are typeset in this paper.

The first example illustrates the interplay between P-, C-, and S-phases in one proof.

Prove: $(G)\quad 36 \in \bigcup_{i\in\mathbb{N}} \{j^2 \mid_{j\in\mathbb{N}} j \geq i \wedge j \leq i+5\}$ under the assumption

$(A)\quad \forall_{m,n}\ n > m \Rightarrow \exists_i\ i \leq n \wedge i \geq m \wedge i \in \mathbb{N}$.

In order to show (G) we have to show

$$(1)\quad \exists_i\ 36 \in \{j^2 \mid_{j\in\mathbb{N}} j \geq i \wedge j \leq i+5\} \wedge i \in \mathbb{N}\ .$$

In order to prove (1) we have to show

$$(2)\quad \exists_i\ \exists_j\ j \geq i \wedge j \in \mathbb{N} \wedge j \leq i+5 \wedge i \in \mathbb{N} \wedge 36 = j^2\ .$$

Since $j := 6$ solves the equational part of (2) it suffices to show

$$(3)\quad \exists_i\ i \in \mathbb{N} \wedge 6 \geq i \wedge 6 \in \mathbb{N} \wedge 6 \leq 5+i\ .$$

Using available computation rules we evaluate (3):

$$(4)\quad \exists_i\ i \leq 6 \wedge i \geq 1 \wedge i \in \mathbb{N}\ .$$

Formula (4), using (A), is implied by:

$$(5)\quad 6 > 1\ .$$

Using available computation rules we evaluate (5):

$$(6)\quad \mathit{True}\ .$$

The derivations of formulae (1) and (2) result from applying `STP` inference rules for membership in a union and membership in a set abstraction, respectively. Reduction of (2) to (3) is accomplished by instantiating j by a solution of a quadratic equation done in `STS`. Simplifications from (3) to (4) and from (5) to (6) were made using available semantic knowledge by `STC` ($6 \in \mathbb{N}$ and $6 > 1$, respectively) and, finally, reduction from (4) to (5) and the detection of proof success were made by standard predicate logic inference rules.

The second example is taken from the set theory section of the TPTP library.
Prove: $(G) \quad B \setminus (C \cap D) = (B \setminus C) \cup (B \setminus D)$.
$\subseteq$: We assume

$$(1) \quad B1 \in B \setminus (C \cap D)$$

and show

$$(2) \quad B1 \in (B \setminus C) \cup (B \setminus D) \ .$$

From (1) we can infer

$$(3) \quad B1 \in B$$

$$(4) \quad B1 \notin C \cap D \ .$$

From (4) we can infer

$$(5) \quad B1 \notin C \vee B1 \notin D \ .$$

In order to prove (2) we may assume

$$(6) \quad B1 \notin B \setminus D$$

and show

$$(7) \quad B1 \in B \setminus C$$

From (6) we can infer

$$(8) \quad B1 \notin B \vee B1 \in D \ .$$

We have to prove (7), thus, we first show:

$$(9) \quad B1 \in B :$$

Formula (9) is true because it is identical to (3).
For proving (7) it still remains to show

$$(10) \quad B1 \notin C :$$

From (3) and (8) we obtain

$$(11) \quad B1 \in D \ .$$

From (5) and (11) we obtain

$$(12) \quad B1 \notin C \ .$$

Formula (10) is true because it is identical to (12).
$\supseteq$: Now we assume (2) and show (1).
From (2) we can infer

$$(13) \quad B1 \in B \setminus C \vee B1 \in B \setminus D \ .$$

We have to prove (1), thus, we first have to show

$$(14) \quad B1 \in B \ .$$

(We skip the proof of (14). It quickly succeeds by case distinction based on (13).)
For proving (1) it still remains to show:

$$(15) \quad B1 \notin C \cap D \ .$$

Assume

$$(16) \quad B1 \in C \cap D$$

From (16) we can infer

$$(17) \quad B1 \in C$$

$$(18) \quad B1 \in D \ .$$

Case (13.1) $B1 \in B \setminus C$:
From (13.1) we can infer

$$(19) \quad B1 \in B$$

$$(20) \quad B1 \notin C \ .$$

(17) and (20) are contradictory.
Case (13.2) $B1 \in B \setminus D$:
From (13.2) we can infer

$$(21) \quad B1 \in B$$

$$(22) \quad B1 \notin D \ .$$

(18) and (22) are contradictory. □

The computation time for proof generation, simplification and display is approx. 6 seconds on a 400 MHz Linux machine. The proof of the same theorem in Otter did not finish within 300 seconds on the same hardware.

References

[1] P. Bernays and A. Fraenkel. *Axiomatic Set Theory*. Studies in Logic and the Foundations of Mathematics. North-Holland Publishing Company, 2 edition, 1968.

[2] B. Buchberger. Mathematics: An Introduction to Mathematics Integrating the Pure and Algorithmic Aspect. Volume I: A Logical Basis for Mathematics. Lecture notes for the mathematics course in the first and second semester at the Fachhochschule for Software Engineering in Hagenberg, Austria, 1996.

[3] B. Buchberger. The PCS Prover in Theorema. In R. Moreno-Diaz, B. Buchberger, and J. Freire, editors, *Proceedings of EUROCAST 2001 (8th International Conference on Computer Aided Systems Theory - Formal Methods and Tools for Computer Science)*, Lecture Notes in Computer Science 2178, 2201, pages 469–478. Springer, Berlin - Heidelberg - New York, 2001.

[4] B. Buchberger and D. Vasaru. The Theorema PCS Prover. Jahrestagung der DMV, Dresden, September 18-22, 2000.

[5] H. Ebbinghaus. *Einführung in die Mengenlehre*. Wissenschaftliche Buchgesellschaft Darmstadt, 2 edition, 1979. ISBN 3-534-06709-6.

[6] A. Formisano. *Theory-based resolution and automated set reasoning*. PhD thesis, Universita degli Studi di Roma "La Sapienza", 2000.

[7] B. Konev and T. Jebelean. Combining Level-Saturation Strategies and Meta-Variables for Predicate Logic Proving in Theorema. In *Proceedings of IMACS ACA 2000, St.Petersburg, Russia*, June 2000.

[8] W. Quine. *Set Theory and its Logic*. Belknap Press of Harvard University Press, Cambridge, Massachusetts, 1963.

[9] B. Russell and A. Whitehead. *Principia Mathematica*. Cambridge University Press, 1910. Reprinted 1980.

[10] G. Takeuti and W. Zaring. *Introduction to Axiomatic Set Theory*. Graduate Texts in Mathematics 1. Springer Verlag, 1971. ISBN 0-387-05302-6.

[11] E. Tomuta. *An Architecture for Combining Provers and its Applications in the Theorema System*. PhD thesis, The Research Institute for Symbolic Computation, Johannes Kepler University, 1998. RISC report 98-14.

[12] D. Vasaru-Dupré. *Automated Theorem Proving by Integrating Proving, Solving and Computing*. PhD thesis, RISC Institute, May 2000. RISC report 00-19.

[13] W. Windsteiger. *A Set Theory Prover in Theorema: Implementation and Practical Applications*[††]. PhD thesis, RISC Institute, May 2001.

[††]http://www.risc.uni-linz.ac.at/people/wwindste/Public/Reports/PhdThesis

Author Index