CENTER FOR
MACHINE PERCEPTION

CZECH TECHNICAL
UNIVERSITY IN PRAGUE

BACHELOR THESIS

Minimal Problem Solver Generator

Pavel Trutman

pavel.trutman@fel.cvut.cz

April 7, 2015

Available at
http://cmp.felk.cvut.cz/~trutmpav/theses/bsc-pavel-trutman.pdf

Thesis Advisor: Ing. Tomáš Pajdla, PhD.

Acknowledge grants here. Use centering if the text is too short.

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
fax +420 2 2435 7385, phone +420 2 2435 7637, www: http://cmp.felk.cvut.cz

Minimal Problem Solver Generator

Pavel Trutman

April 7, 2015

Text of acknowledgements…

Abstract

Text of abstract...

Resumé

Text of resumé. . .

Contents

Abbreviations

AHA! Some optional explanation before the list. Indentation can be set by the command `\setlength{\AbbrvIndent}{5em}`.

1D	one dimension(al)
2D, 3D, …	two dimension(al), three dimension(al), two dimension(al), three dimension(al), two dimension(al), three dimension(al), two dimension(al), three dimension(al), …
AAM	active appearance model
AI	artificial intelligence
ASM	active shape model
B-rep	boundary representation
BBN	Bayesian belief networks

1 Introduction

Here comes introduction.

2 Polynomial system solving

Firstly we review the state of the art algorithms for computing Gröbner basis. Better understanding of these algoritmhs helps us to more efficiently integrate them into polynomial solving algorithms based on Gröbner basis computation.

2.1 Buchberger's Algorithm

Buchberger's Algorithm was the first algorithm for computing Gröbener basis and it was invented by Bruno Buchberger.

2.1.1 First implementation

The first and easy, but very inefficient, implementation of this algorithm says that we can extend a set F of polynomials to a Gröbner basis only by adding all nonzero remainders $\overline{S(f_i, f_j)}^F$ of all pairs from F into F. The pseudocode of this algorithm can be found in [1] on page 87. Disadvatage of this simple algorithm is that computed Gröbner basis are often bigger than necessary.

2.2 F_4 Algorithm

The F_4 Algorithm [2] by Jean-Charles Faugère is an improved version of the Buchberger's Algorithm. The F_4 replaces the classical polynomial reduction found in the Buchberger's Algorithm by the simultaneous reduction of several polynomials. This reduction mechanism is achieved by a symbolic precomputation and by use of sparse linear algebra methods. F_4 speeds up the reduction step by exchanging multiple polynomial divisions for row-reduction of single matrix.

2.2.1 Improved Algorithm F_4

The main function of F_4 Algorithm consists of two parts. The goal of the first part is to initialize the whole algorithm. That means to generate required pairs and initialize future Gröbner basis G. This function takes each polynomial from the input set and calls function *Update* on it, which updates set P of pairs and set G.

Second part of this algorithm generates new polynomials and includes them into the set G. In each iteration it selects same pairs from P using function *Sel*. How to select pairs is an open question. Some selection strategies are described in the section 2.2.6 on page 6.

Then it split each pair into two tuples. First tuple contains first polynomial f_1 from the pair and monomial t_1 such $\mathrm{LM}(t_1 \times f_1) = \mathrm{lcm}(\mathrm{LM}(f_1), \mathrm{LM}(f_2))$. Second tuple is constructed in the same way from the second polynomial from the pair. All tuples from all selected pairs are put into the set L so duplicates are removed.

Then it calls function *Reduction* on the set L and stores result in the set $\tilde{F}^+$. In the end it iterates throught all new polynomials in the set $\tilde{F}^+$ and calls function *Update* on each of them. This generates new pairs into the set P and extends future Gröbner basis G.

This algorithm terminates when the set P of pairs is empty. Then the set G is a Gröbner basis and it is the output of the algorithm.

2.2.2 Function Update

In this algorithm is used standard implementation of Buchberger Criteria **CITE Gebauer and Moller [GM88]**.

2.2.3 Function Reduction

Task of this function is simple, it performs polynomial division using methods of linear algebra.

Input of this function is set L containing tuples of monimial and polynomial, which were made in the main function of the F_4 Algorithm.

First of all this function calls function *Symbolic Preprocessing* on the set L. This returns set F of polynomials ready to reduce. To use linear algebra methods to perform polynomial division we have to put the polynomials into matrix. Each column of the matrix corresponds to a monomial and the columns have to be ordered with respect to used ordering so the right most column corresponds to a monomial "1". On the other hand each row corresponds to a polynomial from the set F. Construction of the matrix is simple: on the (i, j) posistion in the matrix we put coefficient of the term corresponding to j-th monomial from the i-th polynomial from the set F.

If we have constructed matrix like this we can reduce it to a row echelon form using for example Gauss-Jordan elimination. Note that this matrix is typically sparse so we can use sparse linear algebra methods to save computing time and memory. After elimination we can construct resulting polynomials by multiplication of reduced matrix and vector of monomials.

In the end the function returns set of reduced polynomials only with leading monomials which were not amongs polynomials before reduction.

2.2.4 Function Symbolic Preprocessing

In the first part of the function *Symbolic Preprocessing* we get set L of tuples containing monomial and polynomial. These tuples were made from selected pairs. Then are these tuples simplified by function *Simplify* and after multiplying polynomials with corresponding monomials are results put into the set F.

After that the function goes through all monomials in the set F and for each monomial m looks for some polynomial f from G (future Gröbner basis) such $m = m' \times \mathrm{LM}(f)$ where m' is a some monomial. Found polynomial f and monomial m' are after simplification multiplied and put into set F. The goal of this search is to have for each monomial in F some polynomial in F with the same leading monomial. This will ensure that after polynomial division (using linear algebra) all added polynomials will be reduced for G.

2.2.5 Function Simplify

The purpose of the function *Simplify* is obvious. It tries to simplify a polynomial which is product of multiplication of given monomial m and polynomial f.

Function recursively looks for monomial m' and polynomial f' such $\mathrm{LM}(t' \times f') = \mathrm{LM}(t \times f)$. The polynomial f' is selected from all polynomials that has been reduced in previous iterations (sets $\tilde{F}^+$). We select such polynomial f' that total degree of m' is minimal.

This is done do insert into the set F (set of polynomials ready to reduce) polynomials that are mostly reduced and have small number of monomials. This of course speeds up following reduction.

2.2.6 Selection strategy

For the speed of the F_4 Algorithm is very important how to select in each iteration critical pairs from the list of all critical pairs P. This of course depends on the implemetation of the function *Sel*. There are more possible implemetations:

- The easiest implementation is to select all pairs from P. In this case we reduce all criticals pairs at the same time.
- If the function *Sel* selects only one critical pair then the F_4 Algorithm is the Buchberger's Algorithm. In this case the *Sel* function corresponds to the selection strategy in the Buchberger's Algorithm.
- The best function that Faugère has tested is to select all critical pairs with a minimal total degree. Faugère calls this strategy the *normal strategy for* F_4.

2.3 F_5 Algorithm

3 Automatic generator

3.1 Reimplementation

3.2 Multiple eliminations solver

3.3 Removing unnecessary polynomials

3.4 Matrix partitioning

3.5 F4 strategy

4 Experiments

5 Conclusion

Bibliography

[1] David Cox, John Little, and Donald O'Shea. *Ideals, Varieties, and Algorithms : An Introduction to Computational Algebraic Geometry and Commutative Algebra.* Undergraduate Texts in Mathematics. Springer, New York, USA, 2nd edition, 1997. 4

[2] Jean-Charles Faugère. A new efficient algorithm for computing gröbner bases (f_4). *Journal of pure and applied algebra*, 139(1–3):61–88, 7 1999. 4