Changing the ordering of Gröbner Bases with LLL: Case of Two Variables

Abdolali Basiri
Spaces/Lip6/Loria/CNRS/Université Paris VI
8 Rue Capitaine Scott, F-75015 Paris
basiri@calfor.lip6.fr

Jean-Charles Faugère
Spaces/Lip6/Loria/CNRS/Université Paris VI
case 168, 4 pl. Jussieu, F-75252 Paris Cedex 05
jcf@calfor.lip6.fr

ABSTRACT

We present an algorithm for the transformation of a Gröbner basis of an ideal with respect to any given ordering into a Gröbner basis with respect to any other ordering. This algorithm is based on a modified version of the LLL algorithm. The worst case theoretical complexity of this algorithm is not better than the complexity of the FGLM algorithm; but can also give the theoretical complexity with some parameters depending on the size of the output. When the output is small then algorithm is more efficient. We also present a first implementation of the algorithm in Maple. This algorithm is restricted to the case of two variables but works also in positive dimension.

Categories and Subject Descriptors

I.1.2 [**Computing Methodologies**]: Symbolic and Algebraic Manipulation—*Algorithms*; F.2.1 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity—*Computations on polynomials*

General Terms

Algorithms

Keywords

Gröbner basis, LLL, Reduced Lattice basis, Complexity

1. INTRODUCTION

One of the main tools for solving algebraic systems is the computation of Gröbner bases [6, 7, 8, 10]. Gröbner bases for any ordering can be computed in one step with general algorithms [6, 12, 14]. But, in practice, it is well known that it is often much faster to compute: a Gröbner basis G_1 for an ordering $<_1$; then to transform G_1 into another basis for another ordering $<_2$. When the ideal is zero–dimensional, the FGLM algorithm [16] is such an algorithm. The number of arithmetic operations of this algorithm is $O(nN_b^3)$ where n is the number of variables and N_b the number of solutions (with multiplicity). Transforming a Gröbner basis from a total degree to a pure lexicographical ordering is the "natural way" for solving polynomial systems because, from a complexity point of view, the best ordering is the degree–reverse–lexicographical (DRL) one but it is easier to obtain the solutions (either numerically or symbolically) from a Gröbner basis for the lexicographical ordering. More surprisingly, experiments have shown that the other way (from a lexicographical ordering to a total degree ordering) is also very useful in many applications. For instance, to compute a decomposition into prime ideals it is often a good strategy to compute a lexicographical Gröbner basis of the projection on two variables, compute a decomposition of this projection and then to return to a total degree ordering to split the whole ideal (since this is a strong motivation for our algorithm we have included a sketch of this algorithm in section 9).

On the other hand LLL[20] is a well known algorithm for computing short vectors in a lattice; since a Gröbner basis can be seen as the smallest polynomials in an ideal wrt the divisibility of their heading terms it is natural to compare the two algorithms. A possible benefit of LLL is that the complexity of this algorithm is very sensitive to the output; so one can hope that when the result is small the LLL algorithm should be faster than FGLM.

The difficulty in LLL is that it is difficult to "follow" symbolically. A standard algorithm for implementing the arithmetic of Jacobian groups of curves is LLL ([23, 1, 17]); but in [2], we have replaced LLL by FGLM so that we can compute the Gröbner basis "by hand" on a generic input (that is to say with symbolic parameters as coefficients); so we were able to establish explicit formulas.

The plan of the paper is as follows. The section 3 is devoted to presenting the modified LLL algorithm. The section 4 includes also the proof of the correctness of the algorithm. The theoretical complexity of the algorithm is in 5 and the practical behavior of an implementation of the algorithm in Maple can be found in 6. The necessary mathematical notation is reviewed in section 2. For the sake of completeness we have also included a description of a meta algorithm for decomposing ideals in the appendix(section 9) but this part may be omitted.

The name of this algorithm is simply algorithm LLL. In the rest of this paper LLL stands for this modified version of the LLL [20] algorithm.

ISSAC'03, August 3–6, 2003, Philadelphia, Pennsylvania, USA.

2. DEFINITIONS

In this paper we will denote by K a field, by $K[X,Y]$ the ring of polynomials in two variables with coefficients in K and by $I = (f_1, \dots, f_m)$ an ideal in $K[X,Y]$ generated by $f_1, \dots, f_m$.

We recall now some definitions and properties of Gröbner bases and lattice bases that will be used in what follows. We refer to [5, 11] for basic facts and notations.

Given an ideal I we will denote by $(G,<)$ the reduced Gröbner basis with respect to an admissible ordering $<$. We will say that an element $f \in K[X,Y]$ is *reduced* by G if no element $g \in G$ has a leading term that divides some term of f; a Gröbner basis is *reduced* if each of its elements is reduced by the others.

Definition 1 *Let $(G = (g_1, \dots, g_m), <)$ be a reduced Gröbner basis for I, we denote by r_i, the degree of $HT(g_i)$ w.r.t. Y. The indices $(1, \dots, m)$ can be considered in such a way that $r_i \le r_{i+1}$. For any positive integer number D_Y which is not less than $\deg_Y(G)$, let*

$$B_{D_Y}(G) = \{Y^{j_i} g_i | \ 1 \le i \le m, \ \ 0 \le j_i \le r_{i+1} - r_i - 1\}$$

where $r_{m+1} := D_Y$. We denote by $M_{D_Y}(G)$, the $K[X]$ submodule of $K[X,Y]$ generated by $B_{D_Y}(G)$ which is called D_Y-th $K[X]$ module associated to ideal I w.r.t. $<$. In this case $B_{D_Y}(G)$ is called D_Y-th basis of $K[X]$ module associated to ideal I w.r.t. $<$.

Let $\tilde{b}_1, \dots, \tilde{b}_l$ be vectors in $K[X]^{D_Y}$ which are linearly independent over $K(X)$, where l and D_Y are positive integers and $l \le D_Y$. The lattice $L \subset K[X]^{D_Y}$ of rank l spanned by $\tilde{b}_1, \dots, \tilde{b}_l$ is defined as

$$L = \sum_{i=1}^{l} K[X]\tilde{b}_i = \{\sum_{i=1}^{l} \lambda_i \tilde{b}_i | \ \lambda_i \in K[X], \ \ 1 \le i \le l\}.$$

Consider the natural map from $K[X]^{D_Y}$ to $K[X,Y] = K[X][Y]$ which takes the vector $\tilde{v} = (v_1, \dots, v_{D_Y})$ to the polynomial $v = \sum_{j=1}^{D_Y} v_j Y^{j-1}$. Under this map, the lattice $L \subset K[X]^{D_Y}$ corresponds to the $K[X]$-submodule $M(L)$ of $K[X,Y]$ defined by

$$M(L) = \{v = \sum_{j=1}^{D_Y} v_j Y^{j-1} | \ \ \tilde{v} = (v_1, \dots, v_{D_Y}) \in L\}.$$

Let $b_1, \dots, b_l$ be a free basis for the $K[X]$-submodule $M(L)$ of $K[X,Y]$ and let $\tilde{b}_1, \dots, \tilde{b}_l$ be the corresponding basis for the lattice L. We denote by $B = (b_{i,j} Y^{j-1})$ the $l \times D_Y$ matrix where $b_{i,j}$ is the coefficient of Y^{j-1} in the polynomial $b_i = \sum_{j=1}^{D_Y} b_{i,j} Y^{j-1}$. Then we define the determinant $d(M(L))$ of $M(L)$ to be the maximum of the determinant of $l \times l$ sub-matrices of B w.r.t. $<$, and the determinant $d(L)$ of L to be the determinant $d(M(L))$ of $M(L)$. Finally, the orthogonality defect $OD(\tilde{b}_1, \dots, \tilde{b}_l)$ of the basis $\tilde{b}_1, \dots, \tilde{b}_l$ for the lattice L w.r.t. $<$, is defined as

$$HT(b_1) \cdots HT(b_l) - HT(d(L)).$$

Definition 2 *The basis $\tilde{b}_1, \dots, \tilde{b}_l$ is called reduced if*

$$OD(\tilde{b}_1, \dots, \tilde{b}_l) = 0.$$

For $1 \le i \le l$ a i-th *successive minimum* of $M(L)$ w.r.t. $<$ is a minimum element m_i of $M(L)$, such that m_i do not belong to the $K[X]$ submodule of $M(L)$, generated by $m_1, \dots, m_{i-1}$. We remark that m_i is independent of the choice of $m_1, \dots, m_{i-1}$. See [22].

Proposition 3 *Let $\tilde{b}_1, \dots, \tilde{b}_l$ be a reduced basis for a lattice $L \subset K[X]^{D_Y}$ of rank $l \le D_Y$, ordered in such a way that $b_i \le b_j$ for $1 \le i < j \le l$. Then b_i is a i-th successive minimum of $M(L)$ w.r.t. $<$ for $1 \le i \le l$.*

PROOF. See [21]. □

Proposition 4 *Let $\tilde{b}_1, \dots, \tilde{b}_l$ be a basis for a lattice $L \subset K[X]^{D_Y}$ of rank $l \le D_Y$. If the coordinates of the vectors $\tilde{b}_1, \dots, \tilde{b}_l$ can be permuted in such a way that they satisfy*

- $b_i \le b_j$ for $1 \le i < j \le l$
- $b_{i,j} < b_{i,i} \ge b_{i,k}$ for $1 \le i < j \le l, \ \ i < k \le D_Y$

then the basis $\tilde{b}_1, \dots, \tilde{b}_l$ is reduced.

PROOF. See [23]. □

Theorem 5 *Let $(G = (g_1, \dots, g_m), <)$ be a reduced Gröbner basis for I, D_Y a positive integer which is not less than $\deg_Y(G)$ and I_{D_Y} be the set of all polynomials in I whose degree respect to Y is less than D_Y, then $M_{D_Y}(G) = I_{D_Y}$.*

PROOF. It is trivial that $M_{D_Y}(G) \subseteq I_{D_Y}$. If $M_{D_Y}(G) \ne I_{D_Y}$, let h be the minimum polynomials(w.r.t. $<$) in I_{D_Y} which do not belong to $M_{D_Y}(G)$. Let $HT(h) = X^s Y^r$ and for $1 \le i \le m$, $HT(g_i) = X^{s_i} Y^{r_i}$, we can assume $r_i \le r_{i+1}$. Since G is reduced we have $r_i < r_{i+1}$ and $s_i > s_{i+1}$ for $1 \le i \le m-1$ (if $r_i = r_{i+1}$ for some $1 \le i \le m-1$ then $HT(g_i)|HT(g_{i+1})$ or $HT(g_{i+1})|HT(g_i)$, and if $s_i \le s_{i+1}$ then $HT(g_i)|HT(g_{i+1})$ because $r_i < r_{i+1}$). Let

$$i_0 = \max\{i \le m \mid HT(g_i)|HT(h)\}$$

we claim that $i_0 = m$ or $r_{i_0} \le r < r_{i_0+1}$. Otherwise there exist $r_{i_0} < r_{i_0+1} \le r$ then since $s_{i_0+1} < s_{i_0} \le s$ we will have $HT(g_{i_0+1})|HT(h)$ which is a contradiction with choice of i_0.

Thus we have $r - r_{i_0} \le r_{i_0+1} - r_{i_0} + 1$ or $r - r_{i_0} = r - r_m \le D_Y - r_m$ whence there is $b = Y^{r - r_{i_0}} g_{i_0}$ in $B_{D_Y}(G)$ such that $HT(b) = X^{s_{i_0}} Y^r$. Now if we put

$$\tilde{h} = h - \frac{HM(h)}{HM(b)} b = h - \frac{HC(h)}{HC(b)} X^{s - s_{i_0}} b$$

then we will have $\tilde{h}$ belongs to $M_{D_Y}(G)$ and $HT(\tilde{h}) < HT(h)$, thus $\tilde{h} < h$. On the other hand $\tilde{h}$ belongs to I_{D_Y}, which is a contradiction with choice of h. Hence $M_{D_Y}(G) = I_{D_Y}$. □

3. THE ALGORITHM

In this section we present a new version of the LLL algorithm [20] which computes a Gröbner basis for some ordering from the Gröbner base corresponding to another ordering in $K[X,Y]$.

Algorithm 6 (LLL-Paulus)
Input : $(G_{old} = (g_1, \dots, g_m), <_{old})$ a reduced Gröbner basis for I, $<_{new}$, and D_Y a positive integer sufficiently large
Output : $G_{new} = (a_1, \dots, a_l)$ a Gröbner basis for I w.r.t. $<_{new}$

0. $(b_1,\ldots,b_l) \leftarrow$ ModuleBasis$(G_{old}, <_{old}, D_Y)$

1. $k \leftarrow 0$

while $k < l$ **do**

2.1. Choose $i_0 \in \{k+1,\ldots,l\}$ s.t $b_{i_0} = \min_{<_{new}}\{b_i \ : \ k+1 \le i \le l\}$, $swap(\tilde{b}_{k+1}, \tilde{b}_{i_0})$

2.2. Choose $j \in \{1,\ldots,D_Y\}$ s.t $HT_{new}(b_{k+1}) = HT_{new}(b_{k+1,j})$

2.3. **if** $j \le k$ **then**
$\tilde{c} \leftarrow \tilde{b}_{k+1} - \frac{HC_{new}(b_{k+1})}{HC_{new}(a_j)} X^{\deg(\tilde{b}_{k+1,j})-\deg(\tilde{a}_{j,j})} \cdot \tilde{a}_j$
else
$\tilde{c} \leftarrow \tilde{b}_{k+1}$

2.4.1 **if** $HT_{new}(c) = HT_{new}(b_{k+1})$ **then**

- $\tilde{a}_{k+1} \leftarrow \tilde{c}$
- Permute $(k+1,\ldots,D_Y)$ s.t. $HT_{new}(a_{k+1,k+1}) = HT_{new}(a_{k+1})$
- $k \leftarrow k+1$

2.4.2. **if** $HT_{new}(c) <_{new} HT_{new}(b_{k+1})$ **then**

- $p \leftarrow \max\{0 \le s \le k \ : \ a_s \le_{new} c\}$
- **for** $i = k+1$ **down to** $p+2$ **do** $\tilde{b}_i \leftarrow \tilde{a}_{i-1}$
- $\tilde{b}_{p+1} \leftarrow \tilde{c}$
- $k \leftarrow p$

ModuleBasis 7
Input: $(G = (g_1,\ldots,g_m), <)$ a reduced Gröbner basis for I and D_Y, a positive integer sufficiently large
Output: $B_{D_Y}(G) = (b_1,\ldots,b_l)$

- $l \leftarrow D_Y - r_1$
- $k \leftarrow 0$
- Permute the indices $(1,\ldots,m)$ s.t $r_i < r_{i+1}$ where $HT(g_i) = X^{s_i}Y^{r_i}$
- **for** i **from** 1 **to** $m-1$ **do**
  - **for** j **from** 0 **to** $r_{i+1} - r_i - 1$ **do**
    - $k \leftarrow k+1$
    - $b_k \leftarrow Y^j g_i$
- **for** j **from** 0 **to** $D_Y - r_m$ **do**
  - $k \leftarrow k+1$
  - $b_k \leftarrow Y^j g_m$

4. CORRECTNESS

Theorem 8 *Algorithm 6 computes a Gröbner basis G_{new} in $K[X,Y]$ such that $Id(G_{old}) = Id(G_{new})$.*

PROOF. Let D_Y be a positive integer such that

$$D_Y \ge \max\{\deg_Y(G_{old}), \deg_Y(G)\}$$

where G is a Gröbner basis for I w.r.t. $<_{new}$. For example we can assume $D_Y = 2\max\deg(G_{old}) - 1$ if the old ordering is the lexicographical ordering and the new ordering is the degree–reverse–lexicographical ordering, see [9, 18]. Also if the degree of the new ordering w.r.t. Y is at most equal to the degree of the old ordering w.r.t. Y then we can assume $D_Y = \deg_Y(G_{old})$.

Termination: There is a finite number of passages through step 2.4.1 because k is increased by 1. Also there is a finite number of passages through step 2.4.2 because

$$HT(a_1)\cdots HT(a_k)HT(b_{k+1})\cdots HT(b_n)$$

becomes smaller in this step and stays unchanged in the step 2.4.1. Hence the number of passages in the principal loop is finite, and algorithms terminates when $k = l$.

Correctness: It is clear that $B_{D_Y}(G)$ and $(a_1,\ldots,a_l)$ generate the same $K[X]$ submodule M of $K[X,Y]$. By Theorem 5, $M = I_{D_Y}$. On the other hand by Proposition 4, $(\tilde{a}_1,\ldots,\tilde{a}_l)$ is a reduced basis for the lattice L with basis $(\tilde{b}_1,\ldots,\tilde{b}_l)$, because the following invariants are valid before step 2.1:

- $a_i \le a_j$ for $1 \le i < j \le k$
- $a_k \le b_j$ for $k < j \le l$
- $a_{i,j} < a_{i,i} > a_{i,t}$ for $1 \le j < i \le k$ and $i < t \le D_Y$

Hence by Proposition 4, a_i is a i-th successive minimum of M and $HT(a_i) < HT(a_{i+1})$ (otherwise $HT(a_i) = HT(a_{i+1})$ thus $a' = a_{i+1} - a_i \in M$ and $HT(a') < HT(a_{i+1})$ which implies a' is dependent with $a_1,\ldots,a_i$, so $a_{i+1} = a' + a_i$ is also dependent with $a_1,\ldots,a_i$ which is a contradiction with choice of a_{i+1}). Now let f be a polynomial in $I_{D_Y} = M$, then there are $\lambda_1,\ldots,\lambda_l \in K[X]$ such that

$$f = \sum_{j=1}^{l} \lambda_j a_j$$

but for $1 \le i < j \le l$, $HT(\lambda_i a_i) \ne HT(\lambda_j a_j)$ because otherwise there are t_i, t_j s.t. $HT(\lambda_i a_i) = X^{t_i}HT(a_i)$ and $HT(\lambda_j a_j) = X^{t_j}HT(a_j)$, but $HT(a_i) < HT(a_j)$ implies $t_i > t_j$ (if $t_i < t_j$ then $X^{t_i}HT(a_i) < X^{t_j}HT(a_j)$, and if $t_i = t_j$ then $HT(a_i) = HT(a_j)$) hence $a' = X^{t_i - t_j}a_i - a_j \in M$ and $HT(a') < HT(a_j)$ which implies a' is dependent with $a_1,\ldots,a_{j-1}$, so $a_j = a' - X^{t_i-t_j}a_i$ depends with $a_1,\ldots,a_{j-1}$ which is a contradiction with the choice of a_{i+1}. Finally there is a unique $1 \le j \le l$ such that $HT(f) = HT(\lambda_j a_j)$, so $HT(a_j)|HT(f)$ which shows that $(a_1,\ldots,a_l)$ is a Gröbner basis for I w.r.t. $<_{new}$. $\square$

Remark. Unlike FGLM, this algorithm is not limited to zero dimensional ideals.

5. COMPLEXITY

We are now ready to compute the complexity of Algorithm 6 in terms of arithmetical operations in K. By an arithmetical operation in K, we mean addition, subtraction, multiplication or division of two elements of K. We use the notation of Algorithm 6. Denote by

- $d_Y = \max\ \{deg_Y(g_i) \mid 1 \le i \le m\}$
- $D_Y = \max\{d_Y, \deg_Y(G_{new})\}$
- $l = D_Y - r_1$ where $r_1 = \deg_Y HT(g_1)$
- $d_X = \max\ \{deg_X(g_i) \mid 1 \le i \le m\}$

- $D_X = \max\{deg_X(a_i) \mid 1 \le i \le l\}$

Every pass of the main loop consists of $O(l.D_X)$ operations in K for step 2.3. Let us look at, in the worst case, the number of passages in the loop.

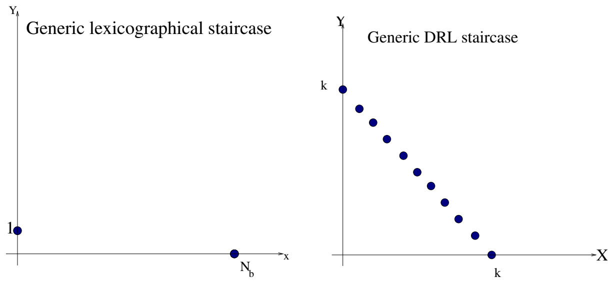
Theorem 9 *The number of passages in the loop of Algorithm 6 is $S = D_Y^2.D_X$.*

PROOF. See appendix(section 8) $\square$

Thus we have proved the following theorem:

Theorem 10 *Algorithm 6 takes $O(D_Y^3.D_X^2)$ arithmetical operations in K to compute a Gröbner basis G_{new} in $K[X,Y]$ such that $Id(G_{old}) = Id(G_{new})$.*

Let I be a zero dimensional ideal and denote by N_b, the dimension of the K vector space $\frac{K[X,Y]}{I}$. Since the complexity of FGLM [16] is $O(N_b^3)$ it is natural to try to express the complexity given by theorem 10 in terms of N_b. A very conservative estimate for D_X, D_Y is: $D_X \le N_b$ and $D_Y \le N_b$; thus Algorithm 6 takes $O(N_b^5)$ arithmetical operations in K to compute a Gröbner basis G_{new} in $K[X,Y]$ such that $Id(G_{old}) = Id(G_{new})$. A more realistic estimate can be done if we consider "generic" ideals; in that case the shape of the lexicographical is (*shape basis [4]*): $[Y - Q(X), P(X)]$ where P (resp. Q) is a univariate polynomial of degree N_b (resp. $N_b - 1$). The staircase (the list of leading monomials of the elements of the Gröbner basis) of such an ideal is plotted in the following figure.



with $N_b = \frac{k(k+1)}{2}$

$$d_Y = 1, d_X = N_b \qquad d_X \approx d_Y \approx \frac{-1+\sqrt{1+8\,N_b}}{2}$$

$$D_Y \approx \frac{-1+\sqrt{1+8\,N_b}}{2}, D_X = N_b$$

$$D_Y^3.D_X^2 = O(N_b^{\frac{7}{2}})$$

Hence, for almost all systems, Algorithm 6 takes $O(N_b^{3.5})$ arithmetical operations in K to compute a Gröbner basis G_{Lexico} in $K[X,Y]$ such that $Id(G_{DRL}) = Id(G_{Lexico})$ (the same is true from Lexico to DRL). The best case for this algorithm is when D_Y is small.

6. EXPERIMENTS

We have implemented Algorithm 6 in *Maple V Release 5*. In tables 1 and 2 we give the timings for some well known examples (Pentium 3 at 800 Mhz) modulo 65521. Some words of caution are necessary: the quality of the computer implementation of Gröbner bases computations may have a dramatic effect on their performance. On the other hand, a Maple implementation of such an algorithm is not an efficient implementation even if it is useful to test the correctness of the algorithm and to give *a rough idea* of its practical behavior. Of course this implementation can not be compared with a low level implementation in C (as in FGb for instance). Another consequence is that we must restrict ourselves to middle size benchmarks. For all this reasons we add also in tables 1 and 2 the number of arithmetical operations, N_b (namely the number of multiplications modulo 65521): this number does not depend on the implementation and so can give an estimate of the intrinsic practical complexity of the algorithm. We add also the values of the parameters N_b, l, d_X, D_X, d_Y and D_Y for each examples.

In tables 1 and 2 we compare performance of the new algorithm on various examples: a first family of well known benchmarks [16, 12, 14] was used; since all this examples have more than two variables we need, first, to *eliminate* $n-2$ variables (see also appendix 9). This can be done easily by computing a Gröbner basis for an appropriate ordering. The second list of examples corresponds to random system in two variables. More precisely rand-d-k is the ideal $Id(y - Q(x), P(x))$ where Q and P are univariate random polynomials of degree d and and weight of k for y in the total degree ordering. All the examples are available on our web page [15].

Since the preliminary version of this paper [3] the implementation in Maple of the LLL algorithm has made substantial progresses (from 3 to 300 times faster). This was done by adding to each entry of the matrix the corresponding leading term (a very slow operation in Maple) and by replacing the basic operation $P \leftarrow P + tQ$ by $P \leftarrow$ (euclidean) remainder of P divided by Q, where P, Q are polynomials and t a monomial.

From tables 1 and 2, a first conclusion is that: this not optimized implementation of LLL is always faster than the standard implementation of FGLM *in Maple.* It is also clear that the best case of the algorithm is when D_Y is very small then the complexity of Algorithm 6 is simply $O(N_b^2)$ (for examples rand-50-17, rand-100-34 ,rand-150-51, rand-200-67 and rand-300-101 D_Y becomes 2).

7. CONCLUSION

We have presented a new version of the LLL algorithm for computing Gröbner bases by changing the ordering. We give a proof and the theoretical complexity of the algorithm. A first implementation in Maple is also presented and the first experimental results are encouraging. An open issue is to generalize this technique to more than two variables.

8. APPENDIX: PROOF OF COMPLEXITY

In every pass of the principal loop, either k increased by 1 or $OD(\tilde{a}_1, \ldots, \tilde{a}_k, \tilde{b}_{k+1}, \ldots, \tilde{b}_l)$ decreases(w.r.t. $<_{new}$), in other words in this case we have

$$c = b_{k+1} - \frac{HC_{new}(b_{k+1})}{HC_{new}(a_j)} X^{\deg(b_{k+1,j}) - \deg(a_{j,j})} \cdot a_j,$$

and $HT(a_1)\cdots HT(a_k)HT(c)HT(b_{k+2})\cdots HT(b_l)$, with respect to $<_{new}$, is less than
$HT(a_1)\cdots HT(a_k)\ HT(b_{k+1})HT(b_{k+2})\cdots HT(b_l)$. Hence

$$\deg_X(HT(c)) \le \deg_X(HT(b_{k+1})) + D_X$$

and

$$\deg_Y(HT(c)) \le D_Y.$$

Note that for two extremes order, the degree reverse lexicographical ordering($<_{DRL}$), and the lexicographical ordering ($<_{Lex}$) we have

- $X^{n_{1,0}} <_{Lex} \cdots <_{Lex} X^{n_{1,1}}$
- $X^{n_{2,0}}Y <_{Lex} \cdots <_{Lex} X^{n_{2,1}}Y$
- $X^{n_{3,0}}Y^2 <_{Lex} \cdots <_{Lex} X^{n_{3,1}}Y^2$

$\vdots$

- $X^{n_{ld_Y,0}}Y^{ld_Y-1} <_{Lex} \cdots <_{Lex} X^{n_{ld_Y,1}}Y^{ld_Y-1}$
- $X^{n_{ld_Y+1,0}}Y^{ld_Y} <_{Lex} \cdots <_{Lex} X^{n_{ld_Y+1,1}}Y^{ld_Y}$.

Thus the number of passages in the loop, in this case, is

$$S = \sum_{i=1}^{ld_Y+1} n_{i,1} - n_{i,0}$$

but $\deg_X(HT(c)) \le \deg_X(HT(b_{k+1})) + D_X$ implies that

$$n_{i,1} \le n_{i+1,0} + D_X \quad \text{for } i = 1, \ldots, ld_Y$$

and hence

$$S \le n_{ld_Y+1,1} + ld_Y D_X - n_{1,0}.$$

Thus the number of passages in the loop, in this case, is bounded by $S = l.d_Y.D_X$.

On the other hand

$$X <_{DRL} Y$$
$$\vdots$$
$$X^{ld_X-1}Y^m <_{\mathrm{DRL}} X^{ld_X-2}Y^{m+1} \cdots <_{\mathrm{DRL}} X^{ld_X-D_Y-1}Y^{m+D_Y}$$
$$X^{ld_X}Y^m <_{\mathrm{DRL}} X^{ld_X-1}Y^{m+1} \cdots <_{\mathrm{DRL}} X^{ld_X-D_Y}Y^{m+D_Y}.$$

Thus the number of passages in the loop, in this case, is bounded by $S = l.d_X.D_Y$.

Hence the number of passages in the loop, in the worst case, is bounded by

$$S = D_Y^2 D_X.$$

9. APPENDIX: META ALGORITHM FOR DECOMPOSING AN IDEAL INTO PRIMES

We give now the sketch of an algorithm to *speed up* the decomposition of an ideal given by a finite set of generators into prime ideals. More precisely if I is an ideal the decomposition into primes is:

$$\sqrt{I} = P_1 \cap \cdots \cap P_k$$

then the output of the algorithm is $[G_1, \ldots, G_k]$ where G_i is the Gröbner basis of P_k for any ordering. The following algorithm is in fact a "meta algorithm": it calls a general decomposition algorithm `DEC` (see for instance [5]) but the idea is to apply this general algorithm `DEC` *after* decomposing as much as possible the initial ideal. Hence, in the worst case, this algorithm is not more efficient than `DEC` but in practice it is often much faster when the dimension is ≤ 1. In [13] we have used a similar method to decompose the cyclic 9 problem (dimension 1 degree 6156).

Input
F a finite subset of $k[x_1, \ldots, x_n]$

Output
a decomposition into primes of the (radical of the) ideal generated by F.

1) Gröbner
G a Gröbner basis of F for a DRL ordering.

2) Elimination
Compute, by changing the ordering, G_e a Gröbner basis for a block ordering $[x_1, \ldots, x_{n-2}]$ $[x_{n-1}, x_n]$. This is an elimination ordering and so $G_e = G_1 \cup G_2$ where $G_2 \subset k[x_{n-1}, x_n]$.

3) Lexico
Compute, by changing the ordering, G_{lex} a lexicographical Gröbner basis of G_2

4) Decompose (2 variables)
Use [19] to compute a decomposition into prime components:

$$\sqrt{Id(G_2)} = Id(P_1) \cap \cdots \cap Id(P_k)$$

(each P_i is a lexicographical Gröbner basis). The main tool to obtain this decomposition is to compute gcd of polynomials (see [19]) so this can be done efficiently.

5) DRL in two variables
For each prime components compute, by change of ordering, a Gröbner for a DRL ordering:

$$P_i' = \text{Gröbner}(P_i, DRL)\ i = 1, \ldots, k$$

6) DRL

$$G_i' = \text{Gröbner}(G_e \cup P_i', DRL)\ i = 1, \ldots, k$$

7) Decompose (general)
For all G_i' compute a decomposition into primes of G_i', and return:

$$\mathtt{DEC}(G_1') \cup \cdots \cup \mathtt{DEC}(G_k')$$

Key points for the efficiency of the algorithm are steps 1, 3 and 5. Steps 3 and 5 can be done with the algorithm LLL presented in this paper. Ultimately, the decomposition into irreducible factors of an ideal is done by factorization of univariate polynomials; hence we need to project the ideal that is to say computing a Gröbner basis for an *elimination order*; on the other hand it is much more efficient to represent ideals (to compute the intersection of ideals for instance) by Gröbner bases for a *total degree* odering (DRL). Thus it is necessary to be able to change efficiently the internal representation of an ideal by changing the ordering of the Gröbner basis describing the ideal. The efficiency of this "meta algorithm" was demonstrated in [13] by computing a decomposition into primes of Cyclic 9 and 10.

Acknowledgments: We would like to thank Guillaume Hanrot for our motivation for studying LLL and its application in Gröbner bases. We thank Andreas Enge and Nicolas Gürel for interesting discussions on LLL. We thank David Cox for critically and carefully reading the manuscript and making many useful remarks.

10. REFERENCES

[1] S. Arita. Algorithms for computations in Jacobian group of C_{ab} curve and their application to discrete-log based public key cryptosystems. *IEICE Transactions*, J82-A(8):1291–1299, 1999. In Japanese. English translation in the proceedings of the Conference on The Mathematics of Public Key Cryptography, Toronto 1999.

[2] A. Basiri, A. Enge, J.-C. Faugère, and N. Gürel. Fast arithmetics for superelliptic cubics. Extended version, available from http://www.lix.polytechnique.fr/-Labo/Andreas.Enge/vorabdrucke/super.ps, 2002.

[3] A. Basiri and J.-C. Faugère. Changing the ordering of gröbner bases with LLL: Case of two variables. Technical report, Loria, http://www.inria.fr/rrrt/rr-4746.html, February 2003.

[4] E. Becker, T. Mora, M. G. Marinari, and C. Traverso. The shape of the shape lemma. In *Proceedings of the international symposium on Symbolic and algebraic computation*, pages 129–133. ACM Press, 1994.

[5] T. Becker and V. Weispfenning. *Gröbner bases.* Springer-Verlag, NewYork-Berlin-Heidelberg, 1993.

[6] B. Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal.* PhD thesis, Innsbruck, 1965.

[7] B. Buchberger. An algorithmical criterion for the solvability of algebraic systems. *Aequationes Mathematicae*, 4(3):374–383, 1970. (German).

[8] B. Buchberger. A criterion for detecting unnecessary reductions in the construction of gröbner basis. In *Proc. EUROSAM 79*, volume 72 of *Lecture Notes in Computer Science*, pages 3–21. Springer Verlag, 1979.

[9] B. Buchberger. A note on the complexity of constructing Gröbner bases. In *van Hulzen J.A. (ed.) EUROCAL '83, European Computer Algebra Conference*, volume 162 of *Lecture Notes in Computer Science*, pages 137–145. Springer, 1983.

[10] B. Buchberger. Gröbner bases : an algorithmic method in polynomial ideal theory. In R. P. Company, editor, *Recent trends in multidimensional system theory*, chapter 6, pages 184–232. Bose, 1985.

[11] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms.* Undergraduate Texts in Mathematics. Springer-Verlag, 1996. second edition.

[12] J.-C. Faugère. A new efficient algorithm for computing gröbner bases (f4). *Journal of Pure and Applied Algebra*, 139(1–3):61–88, June 1999.

[13] J.-C. Faugère. How my computer find all the solutions of cyclic 9. In *Lecture Notes Series on Computing*, volume 9, pages 1–12. World Scientific Publishing Co., 2001.

[14] J.-C. Faugère. A new efficient algorithm for computing gröbner bases without reduction to zero f5. In T. Mora, editor, *Proceedings of ISSAC*, pages 75–83. ACM Press, July 2002.

[15] J.-C. Faugère. Examples for LLL. Technical report, Lip6, http://calfor.lip6.fr/~jcf/Papers/LLL/index.html, April 2003.

[16] J.-C. Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient computation of zero-dimensional Gröbner bases by change of ordering. *Journal of Symbolic Computation*, 16:329–344, 1993.

[17] R. Harasawa and J. Suzuki. Fast Jacobian group arithmetic on C_{ab} curves. In W. Bosma, editor, *Algorithmic Number Theory — ANTS-IV*, volume 1838 of *Lecture Notes in Computer Science*, pages 359–376, Berlin, 2000. Springer-Verlag.

[18] D. Lazard. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In *van Hulzen J.A. (ed.) EUROCAL '83, European Computer Algebra Conference*, volume 162 of *Lecture Notes in Computer Science*, pages 146–157. Springer-Verlag, 1983.

[19] D. Lazard. Ideal bases and primary primary decomposition:case of two variables. *Journal of Symbolic Computation*, 1(3):261–270, September 1985.

[20] A. Lenstra, H. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:515–534, 1982.

[21] A. K. Lenstra. Factoring multivariate polynomials over finite fields. *J. Computer and System Sciences*, 30(2):235–248, 1985.

[22] K. Mahler. An analogue of minkowski's geometry of numbers in a field of series. In *Annals of Math*, volume 42, pages 488–522, 1941.

[23] S. Paulus. Lattice basis reduction in function fields. In J. P. Buhler, editor, *Algorithmic Number Theory — ANTS-III*, volume 1423 of *Lecture Notes in Computer Science*, pages 567–575, Berlin, 1998. Springer-Verlag.

DRL $\to$ Lex	Nb of $*$ in LLL	N_b	FGLM (sec)	LLL (sec)	l	d_X	D_X	$d_Y = D_Y$
benchmarkD1	575	48	15.4	.15	3	25	48	2
Cyclic5	37	55	2.1	.25	9	12	15	8
UteshevBikker	$19,721$	36	43.6	1.5	9	8	36	8
Fabrice24	$22,799$	40	138.3	1.6	9	9	40	8
dessin2	$28,821$	42	145.2	1.8	9	9	42	8
dessin1	$45,357$	46	305.7	2.4	10	10	46	9
benchmarki1	$188,997$	66	579.4	6.4	12	11	66	11
cyclic6	$50,938$	126	421.8	3.15	13	24	48	12
katsura7	$2,053,674$	128	$> 12,000$	40.2	16	16	128	15
katsura8	$45,992,680$	256	$> 12,000$	356.8	23	23	241	22
rand-50-17	$1,632$	50	10.5	.1	3	34	50	2
rand-100-34	$6,666$	100	141.4	.3	3	67	100	2
rand-150-51	$14,798$	150	834.8	.5	3	101	150	2
rand-200-67	$30,822$	200	4261	.8	3	134	200	2
rand-300-101	$69,300$	300	> 4000	1.4	3	201	300	2
rand-50-1	$52,758$	50	364.5	2.6	10	10	50	9
rand-100-1	$792,333$	100	> 5000	17.5	14	14	100	13
rand-150-1	$4,196,783$	150	$> 20,000.$	62.6	17	17	150	16
rand-200-1	$17,950,766$	200	$> 20,000.$	157.2	20	20	200	19
rand-300-1	$106,495,461$	300	$> 20,000.$	766.7	25	25	300	24

Table 1: Comparison FGLM/LLL (from DRL to Lex) modulo p.

Lex $\to$ DRL	Nb of $*$ in LLL	N_b	FGLM (sec)	LLL (sec)	l	$d_X = D_X$	d_Y	D_Y
benchmarkD1	$1,200$	48	8.6	.2	3	48	1	2
cyclic5	307	55	2.4	.5	9	15	7	8
UteshevBikker	$46,111$	36	186.5	7.3	9	36	1	8
Fabrice24	$56,550$	40	245.4	8.8	9	40	1	8
dessin2	$62,588$	42	291.0	9.3	9	42	1	8
dessin1	$94,325$	46	522.1	13.4	10	46	1	9
benchmarki1	$285,970$	66	$2,847.7$	35.3	12	66	1	11
cyclic6	$11,070$	126	75.1	7.1	13	48	6	12
katsura7	$1,941,523$	128	$> 12,000.$	192.2	16	128	2	15
katsura8	$23,187,086$	256	$> 12,000.$	1927.	23	241	16	22
rand-50-17	$4,896$	50	15.1	.35	3	50	1	2
rand-100-34	$19,760$	100	216.7	.9	3	100	1	2
rand-150-51	$44,394$	150	1486.1	1.7	3	150	1	2
rand-200-67	$79,596$	200	8090.	2.8	3	200	1	2
rand-300-101	$178,794$	300	> 8000	5.4	3	300	1	2
rand-50-1	$110,774$	50	339.3	15.1	10	50	1	9
rand-100-1	$905,447$	100	$> 6,000.$	100.6	14	100	1	13
rand-150-1	$3,052,423$	150	$> 50,000$	310.5	17	150	1	16
rand-200-1	$7,646,326$	200	$> 50,000$	711.9	20	200	1	19
rand-300-1	$27,156,881$	300	$> 50,000$	2647.5	25	300	1	24

Table 2: Comparison FGLM/LLL (from Lex to DRL) modulo p.