Polynomial Systems Solving by Fast Linear Algebra

Jean-Charles Faugère, Pierrick Gaudry, Louise Huot, Guénaël Renault

Polynomial Systems Solving by Fast Linear Algebra.

Jean-Charles Faugère[†] Pierrick Gaudry[‡] Louise Huot[†] Guénaël Renault[†]

Abstract

Polynomial system solving is a classical problem in mathematics with a wide range of applications which make its complexity a central study in theoretical computer science. Depending on the context, solving has different meanings. In order to stick to the most general case, we consider a representation of the solutions from which one can easily recover the exact solutions or a certified approximation of them. Under generic assumption, such a representation is given by the lexicographical Gröbner basis of the system and consists of a set of univariate polynomials. The best known algorithm for computing the lexicographical Gröbner basis is in $O(nD^3)$ arithmetic operations where n is the number of variables and D the number of solutions of the system. We show that this complexity can be decreased to $\widetilde{O}(D^\omega)$ where $2 \leq \omega < 2.3727$ is the exponent in the complexity of multiplying two dense matrices and the notation $\widetilde{O}$ means that we neglect logarithmic factors. To achieve this result we propose new algorithms which rely on fast linear algebra. When the degree of the equations are bounded we propose a deterministic algorithm. In the unbounded case we present a Las Vegas algorithm.

1 Introduction

Context. Polynomial systems solving is a classical problem in mathematics. It is not only an important problem on its own, but it also has a wide spectrum of applications. It spans several research disciplines such as coding theory [14,32], cryptography [9,26], computational game theory [13,40], optimization [24], *etc*. The ubiquitous nature of the problem positions the study of its complexity at the center of theoretical computer science. *Exempli gratia*, in the context of computational geometry, a step of the algorithm by Safey el Din and Schost [2], the first algorithm with better complexity than the one by Canny [11] for solving the road map problem, depends on solving efficiently polynomial systems. In cryptography, the recent breakthrough algorithm due to Joux [26] for solving the discrete logarithm problem in finite fields of small characteristic heavily relies on the same capacity. However, depending on the context, *solving a polynomial system* has different meanings. If we are working over a finite field, then *solving* generally means that we enumerate all the possible solutions lying in this field. On the other hand, if the field is of characteristic zero, then *solving* might mean that we approximate the real (complex) solutions up to a specified precision. Therefore, an algorithm for solving polynomial systems should provide an output that is valid in all contexts. In this paper we present an efficient algorithm to tackle the PoSSo (*Polynomial Systems Solving*) problem, the ouput of which is a representation of the roots suitable in all the cases. The precise definition of the problem is as follows:

Problem 1 (PoSSo). *Let $\mathbb{K}$ be the rational field $\mathbb{Q}$ or a finite field $\mathbb{F}_q$. Given a set of polynomial equations with a finite number of solutions which are all simple*

$$\mathcal{S} : \{f_1 = \cdots = f_s = 0\}$$

[†]UPMC, Université Paris 06; INRIA, Paris Rocquencourt Center; PolSys Project, LIP6/CNRS; UMR 7606, France; Email addresses: Jean-Charles.Faugere@inria.fr, {Louise.Huot,Guenael.Renault}@lip6.fr

[‡]Université de Lorraine; LORIA, Lorraine; CARAMEL Project, LORIA/CNRS; UMR 7503, France; Email address: Pierrick.Gaudry@loria.fr

with $f_1, \ldots, f_s \in \mathbb{K}[x_1, \ldots, x_n]$, find a univariate polynomial representation of the solutions of $\mathcal{S}$ i.e. $h_1, \ldots, h_n \in \mathbb{K}[x_n]$ such that the system $\{x_1 - h_1 = \cdots = x_{n-1} - h_{n-1} = h_n = 0\}$ have the same solutions as $\mathcal{S}$.

It is worth noting that enumerating the solutions in a finite field or approximating the solutions in the characteristic zero case can be easily done once the underlying PoSSo problem is solved. Actually, from a given univariate polynomial representation $\{x_1 - h_1 = \cdots = x_{n-1} - h_{n-1} = h_n = 0\}$ one just have to find the (approximated) roots of the univariate polynomial h_n. The algorithms to compute such roots have their complexities in function of D, the degree of h_n, well handle and in general they are negligible in comparison to the cost of solving the PoSSo problem. Note that D is also the total number of solutions of the polynomial system. For instance, if $\mathbb{K} = \mathbb{F}_q$ is a finite field, the enumeration of the roots lying in $\mathbb{F}_q$ of h_n can be done in $\widetilde{O}(D)$ arithmetic operations where the notation $\widetilde{O}$ means that we neglect logarithmic factors in q and D, see [42]. In the characteristic zero case, finding an approximation of all the real roots of h_n can also be done in $\widetilde{O}(D)$ where, in this case, we neglect logarithmic factors in D, see [37].

A key contribution to the PoSSo problem is the multivariate resultant introduced by Macaulay in the beginning of the 20th century [33]. The next major achievement on PoSSo appeared in the 1960s when Buchberger introduced, in his PhD thesis, the concept of Gröbner bases and the first algorithm to compute them. Since then, Gröbner bases have been extensively studied (see for instance [4, 12, 30, 40]) and have become a powerful and a widely used tool to solve polynomial systems. A major complexity result related to the PoSSo problem has been shown by Lakshman and Lazard in [30] and states that this problem can be solved in a simply exponential time in the maximal degree of the equations. As the number of solutions can be bounded by an exponential in this degree (Bézout's bound), this result yields the first step toward a polynomial complexity in the number of solutions for the PoSSo problem. Whereas for the particular case of approximating or computing a rational parametrization of all the solutions of a polynomial systems with coefficients in a field of characteristic zero there exist algorithms with sub-cubic complexity in D ($\widetilde{O}(12^n D^2)$ for the approximation, see [36], and $O\left(n2^n D^{\frac{5}{2}}\right)$ for the rational parametrization, see [7]). In the best of our knowledge, for the complexity of computing a univariate polynomial representation of the solutions, there is no better bound than $O(nD^3)$. The main goal of this paper is to pass over this theoretical barrier and thus providing the first algorithm with sub-cubic complexity in D to solve the PoSSo problem.

Related works. In order to reach this goal we develop new algorithms in Gröbner basis theory. Let $\mathcal{S}$ be a polynomial system in $\mathbb{K}[x_1, \ldots, x_n]$ verifying the hypothesis of Problem 1, *i.e.* with a finite number of solutions in an algebraic closure of $\mathbb{K}$ which are all simple. A Gröbner basis is to $\mathcal{S}$ what row echelon form is to a linear system. For a fixed monomial ordering, given a system of polynomial equations, its associated Gröbner basis is unique after normalization. From an algorithmic point of view, monomial orderings may differ: some are attractive for the efficiency whereas some others give rise to a more structured output. Hence, the fastest monomial ordering is usually the degree reverse lexicographical order, denoted DRL. However, in general, a DRL Gröbner basis does not allow to list the solutions of $\mathcal{S}$. An important ordering which provides useful outputs is the lexicographical monomial ordering, denoted LEX in the sequel. Actually, for a characteristic 0 field or with a sufficiently large one, up to a linear change of the coordinates, a Gröbner basis for the LEX ordering of the polynomial system $\mathcal{S}$ gives a univariate polynomial representation of its solutions [23,29]. That is to say, computing this Gröbner basis is equivalent to solving the PoSSo problem 1. It is usual to define the following: the ideal generated by $\mathcal{S}$ is said to be in *Shape Position* when its LEX Gröbner basis is a polynomial representation of its solutions (*i.e.* one do not need to apply any linear change of coordinates). In a first part of this paper, we will avoid the consideration of the probabilistic choice of the linear change of coordinates in order to be in Shape Position, thus we assume the following hypothesis.

Hypothesis 1. *Let $\mathcal{S} \subset \mathbb{K}[x_1, \ldots, x_n]$ be a polynomial system with a finite number of solutions which are all simple. Its associated LEX Gröbner basis is in Shape Position.*

From a DRL Gröbner basis, one can compute the corresponding LEX Gröbner basis by using a change of ordering algorithm. Consequently, when the associated LEX Gröbner basis of the system $\mathcal{S}$ is in *Shape Position i.e.* $\mathcal{S}$ verifies Hypothesis 1 the usual and most efficient algorithm is first to compute a DRL Gröbner basis. Then, the LEX Gröbner basis is computed by using a change of ordering algorithm. This is summarized in Algorithm 1.

Algorithm 1: Solving polynomial systems

Input : A polynomial system $\mathcal{S} \subset \mathbb{K}[x_1, \ldots, x_n]$ which verifies Hypothesis 1.
Output: The LEX Gröbner basis of $\mathcal{S}$ *i.e.* the univariate polynomial representation of the solutions of $\mathcal{S}$.

Computing the DRL Gröbner basis of $\langle S \rangle$;
From the DRL Gröbner basis, computing the LEX Gröbner basis of $\langle S \rangle$;
return The LEX Gröbner basis of $\mathcal{S}$;

The first step of Algorithm 1 can be done by using F_4 [16] or F_5 [17] algorithms. The complexity of these algorithms for *regular systems* is well-handled. For the homogeneous case, the regular property for a polynomial system $\{f_1, \ldots, f_s\} \subset \mathbb{K}[x_1, \ldots, x_n]$ is a generic property which implies that for all $i \in \{2, \ldots, s\}$, the polynomial f_i does not divide zero in the quotient ring $\mathbb{K}[x_1, \ldots, x_n]/\langle f_1, \ldots, f_{i-1} \rangle$. There is an analogous definition for the affine case, see Definition 4. For the particular case of the DRL order, computing a DRL Gröbner basis of a regular system in $\mathbb{K}[x_1, \ldots, x_n]$ with equations of same degree, d, can be done in $O(d^{\omega n})$ arithmetic operations (see [1, 31]). Moreover, the number of solutions D of the system can be bounded by d^n by using the Bézout's bound. Since, this bound is generically (*i.e.* almost always) reached *i.e.* $D = d^n$, computing a DRL Gröbner basis can be done in $O(D^\omega)$ arithmetic operations. Hence, in this case the first step of Algorithm 1 has a polynomial arithmetic complexity in the number of solutions with exponent ω.

The second step of Algorithm 1 can be done by using a change of ordering algorithm. In 1993, Faugère *et al.* showed in [19] that change of ordering for zero dimensional ideals is closely related to linear algebra. Indeed, they proposed a change of ordering algorithm, denoted FGLM in the literature, which proceeds in two stages. Let $\mathbb{G}_{>_1}$ be the given Gröbner basis w.r.t. the order $>_1$ of an ideal in $\mathbb{K}[x_1, \ldots, x_n]$. First, we need for each $i \in \{1, \ldots, n\}$ a matrix representation, T_i, of the linear map of $\mathbb{K}[x_1, \ldots, x_n]/\langle \mathbb{G}_{>_1} \rangle \rightarrow \mathbb{K}[x_1, \ldots, x_n]/\langle \mathbb{G}_{>_1} \rangle$ corresponding to the multiplication by x_i. The matrix T_i is called multiplication matrix by x_i. These matrices are constructed by computing $O(nD)$ matrix-vector products (of size $D \times D$ times $D \times 1$). Hence, the first stage of FGLM algorithm (Algorithm 2) has an arithmetic complexity bounded by $O(nD^3)$. Once all the multiplication matrices are computed, the second Gröbner basis w.r.t. the new monomial order $>_2$ is recovered by testing linear dependency of $O(nD)$ vectors of size $D \times 1$. This can be done in $O(nD^3)$ arithmetic operations. This algorithm is summarized in Algorithm 2. In consequence, solving regular zero-dimensional systems can be done in $O(nD^3)$ arithmetic operations and change of ordering appears as the bottleneck of PoSSo.

Fast Linear Algebra. Since the second half of the 20th century, an elementary issue in theoretical computer science was to decide if most of linear algebra problems can be solved by using fast matrix multiplication and consequently bound their complexities by that of multiplying two dense matrices *i.e.* $O(m^\omega)$ arithmetic operations where $m \times m$ is the size of the matrix and $2 \leq \omega < 2.3727$. This upper bound for ω was obtained by Vassilevska Williams in [41]. For instance, Bunch and Hopcroft showed in [10] that the

Algorithm 2: FGLM

Input : The Gröbner basis w.r.t. $>_1$ of an ideal $\mathcal{I}$.

Output: The Gröbner basis w.r.t. $>_2$ of $\mathcal{I}$.

1 Computing the multiplication matrices $T_1, \ldots, T_n$; *// $O(nD)$ matrix-vector products*

2 From $T_1, \ldots, T_n$ computing the Gröbner basis of $\mathcal{I}$ w.r.t. $>_2$; *// $O(nD)$ linear dependency tests*

inverse or the triangular decomposition can be done by using fast matrix multiplication. Baur and Strassen investigated the determinant in [3]. The case of the characteristic polynomial was treated by Keller-Gehrig in [27]. Although that the link between linear algebra and the change of ordering has been highlighted for several years, relating the complexity of the change of ordering with fast matrix multiplication complexity is still an open issue.

Main results. The aim of this paper is then to give an initial answer to this question in the context of polynomial systems solving *i.e.* for the special case of the DRL and LEX orderings. More precisely, our main results are summarized in the following theorems. First we present a *deterministic* algorithm computing the univariate polynomial representation of a polynomial system verifying Hypothesis 1 and whose equations have bounded degree.

Theorem 1.1. *Let $\mathcal{S} = \{f_1, \ldots, f_n\} \subset \mathbb{K}[x_1, \ldots, x_n]$ be a polynomial system verifying Hypothesis 1 and let $\mathbb{K}$ be the rational field $\mathbb{Q}$ or a finite field $\mathbb{F}_q$. If the sequence $(f_1, \ldots, f_n)$ is a regular sequence and if the degree of each polynomial f_i ($i = 1, \ldots, n$) is uniformly bounded by a fixed integer d then there exists a deterministic algorithm solving Problem 1 in $\widetilde{O}(d^{\omega n} + D^\omega)$ where the notation $\widetilde{O}$ means that we neglect logarithmic factors in D and polynomial factors in n and d.*

Then we present a *Las Vegas* algorithm extending the result of Theorem 1.1 to polynomial systems not necessarily verifying Hypothesis 1 and whose equations have non fixed degree.

Theorem 1.2. *Let $\mathcal{S} = \{f_1, \ldots, f_n\} \subset \mathbb{K}[x_1, \ldots, x_n]$ be a polynomial system and let $\mathbb{K}$ be the rational field $\mathbb{Q}$ or a finite field $\mathbb{F}_q$. If the sequence $(f_1, \ldots, f_n)$ is a regular sequence where the degree of each polynomial is uniformly bound by a non fixed parameter d then there exists a Las Vegas algorithm solving Problem 1 in $\widetilde{O}(d^{\omega n} + D^\omega)$ arithmetic operations; where the notations $\widetilde{O}$ means that we neglect logarithmic factors in D and polynomial factors in n.*

If $\mathbb{K} = \mathbb{Q}$ the probability of success of the algorithm mentioned in Theorem 1.2 is 1 while in the case of a finite field $\mathbb{F}_q$ of characteristic p, the success of the algorithm depends on the size of p and q, see Section 7.2.

As previously mentioned, the Bézout's bound allows to bound D by d^n and generically this bound is reached i.e. $D = d^n$. By consequence, Theorem 1.1 (respectively Theorem 1.2) means that if the equations have fixed (respectively non fixed) degree then there exists a deterministic (respectively a Las Vegas) algorithm computing the univariate polynomial representation of generic polynomial systems in $\widetilde{O}(D^\omega)$ arithmetic operations.

To the best of our knowledge, these complexities are the best ones for solving the PoSSo Problem 1. For example, in the case of field of characteristic zero, under the same hypotheses as in Theorem 1.1, one can now compute a univariate polynomial representation of the solutions in $\widetilde{O}(D^\omega)$ without assuming that the multiplicative structure of $\mathbb{K}[x_1, \ldots, x_n]$ is known. This can be compared to the method in [7] which, assuming the multiplicative structure of the quotient ring known, computes a parametrization of the solutions in $O\left(n2^n D^{\frac{5}{2}}\right)$. Noticing that under the hypotheses of Theorem 1.1, n is of the order of $\log_2(D)$ and the algorithm in [7] has a complexity in $\widetilde{O}\left(D^{\frac{7}{2}}\right)$.

Outline of the algorithms. In 2011, Faugère and Mou proposed in [21] another kind of change of ordering algorithm to take advantage of the sparsity of the multiplication matrices. Nevertheless, when the multiplication matrices are not sparse, the complexity is still in $O(D^3)$ arithmetic operations. Moreover, these complexities are given assuming that the multiplication matrices have already been computed and the authors of [21] do not investigate their computation whose complexity is still in $O(nD^3)$ arithmetic operations. In FGLM, the matrix-vectors products (respectively linear dependency tests) are intrinsically sequential. This dependency implies a sequential order for the computation of the matrix-vectors products (respectively linear dependency tests) on which the correctness of this algorithm strongly relies. Thus, in order to decrease the complexity to $\widetilde{O}\left(D^{\omega}\right)$ we need to propose new algorithms.

To achieve result in Theorem 1.1 we propose two algorithms in $\widetilde{O}(D^{\omega})$, each of them corresponding to a step of the Algorithm 2.

We first present an algorithm to compute multiplication matrices assuming that we have already computed a Gröbner basis $\mathbb{G}$. The bottleneck of the existing algorithm [19] came from the fact that nD normal forms have to be computed in a sequential order. The key idea is to show that we can compute *simultaneously* the normal form of all monomials *of the same degree* by computing the row echelon form of a well chosen matrix. Hence, we replace the nD normal form computations by $\log_2(D)$ (we iterate degree by degree) row echelon forms on matrices of size $(n\,D) \times (nD + D)$. To compute simultaneously these normal forms we observe that if r is the normal form of a monomial m of degree $d-1$ then $m - r$ is a polynomial in the ideal of length at most $D+1$; then we generate the Macaulay matrix of all the products $x_i m - x_i r$ (for i from 1 to n) together with the polynomials g in the Gröbner basis $\mathbb{G}$ of degree exactly d. We recall that the Macaulay matrix of some polynomials [31, 33] is a matrix whose rows consist of the coefficients of these polynomials and whose columns are indexed with respect to the monomial ordering. Computing a row echelon form of the concatenation of all the Macaulay matrices in degree less or equal to d enable us to obtain all the normal forms of all monomials of degree d. This yields an algorithm to compute the multiplication matrices of arithmetic complexity $O(dn^{\omega}D^{\omega})$ where d is the maximal degree of the polynomials in $\mathbb{G}$; note that this algorithm can be seen as a redundant version of F_4 or F_5.

In order to prove Theorem 1.2 we use the fact that, in a generic case, only the multiplication matrix by the *smallest variable* is needed. Surprisingly, we show (Theorem 7.1) that, in this generic case, *no arithmetic* operation is required to build the corresponding matrix. Moreover, for non generic polynomial systems, we prove (Corollary 3) that a generic linear change of variables bring us back to this case.

The second algorithm (step 2 of Algorithm 2) we describe is an adaptation of the algorithm given in [21] when the ideal is in *Shape Position*. Once again only the multiplication matrix by the *smallest variable* is needed in this case. When the multiplication matrix T of size $D \times D$ is dense, the $O(D^3)$ arithmetic complexity in [21] came from the $2D$ matrix-vector products $T^i\mathbf{r}$ for $i = 1, \ldots, 2D$ where $\mathbf{r}$ is a column vector of size D. To decrease the complexity we follow the Keller-Gehrig algorithm [27]: first, we compute $T^2, T^4, \ldots, T^{2^{\lceil \log_2 D \rceil}}$ using binary powering; second, all the products $T^i\mathbf{r}$ are recovered by computing $\log_2 D$ matrix multiplications. Then, in the Shape Position case, the n univariate polynomials of the lexicographical Gröbner basis are computed by solving n structured linear systems (Hankel matrices) in $O(nD\log_2^2(D))$ operations. We thus obtain a change of ordering algorithm (DRL to LEX order) for *Shape Position* ideals whose complexity is in $O\left(\log_2(D)\left(D^{\omega} + n\log_2(D)D\right)\right)$ arithmetic operations.

Organization of the paper. The paper is organized as follows. In Section 2 we first introduce some required notations and backgrounds. Then, an algorithm to compute the LEX Gröbner basis given the multiplication matrices is presented in Section 3. Next, we describe the algorithm to compute multiplication matrices in Section 4. Afterwards, their complexity analysis are studied in Section 5 where we obtain Theorem 1.1. Finally, in Section 7 we show how to deduced (*i.e.* without any costly arithmetic operation) the multiplication matrix by the smallest variable. According to this construction we propose another algorithm for polynomial systems solving which allows to obtain the result in Theorem 1.2.

The authors would like to mention that a preliminary version of this work was published as a poster in the ISSAC 2012 conference [18].

2 Notations and preliminaries

Throughout this paper, we will use the following notations. Let $\mathbb{K}$ denote a field (for instance the rational numbers $\mathbb{Q}$ or a finite field $\mathbb{F}_q$ of characteristic p), and $\mathbb{A} = \mathbb{K}[x_1, \dots, x_n]$ be the polynomial ring in n variables with $x_1 > \cdots > x_n$. Let $\mathcal{I}$ be an ideal of $\mathbb{A}$; once a monomial ordering $<$ is fixed, a reduced Gröbner basis $\mathbb{G}_<$ of $\mathcal{I}$ w.r.t. $<$ can be computed. Moreover, we always consider reduced Gröbner basis so henceforth, we omit the adjective "reduced". For instance, $\mathbb{G}_{\text{drl}}$ (resp. $\mathbb{G}_{\text{lex}}$) denotes the Gröbner basis of $\mathcal{I}$ w.r.t. the DRL order (resp. the LEX order). A monomial of $\mathbb{K}[x_1, \dots, x_n]$ is a product of powers of variables and a term is a product of a monomial and a coefficient in $\mathbb{K}$. We denote by $\text{LT}_<(f)$ the leading term of f w.r.t. the monomial ordering $<$.

Definition 1 (Zero-dimensional ideal)**.** *Let $\mathcal{I}$ be an ideal of $\mathbb{A}$. If $\mathcal{I}$ has a finite number of solutions, counted with multiplicities in an algebraic closure of $\mathbb{K}$, then $\mathcal{I}$ is said to be zero-dimensional. This number, denoted by D, is also the degree of the ideal $\mathcal{I}$. If $\mathcal{I}$ is zero-dimensional, then the residue class ring $V_\mathcal{I} = \mathbb{A}/\mathcal{I}$ is a $\mathbb{K}$-vector space of dimension D.*

From $\mathbb{G}_<$ one can deduced a vector basis of $V_\mathcal{I}$. Indeed, the canonical vector basis of $V_\mathcal{I}$ is $B = \{1 = \epsilon_1 < \cdots < \epsilon_D\}$ where ϵ_i are irreducible monomials (that is to say for all $i \in \{1, \dots, D\}$, there is no $g \in \mathbb{G}_<$ such that $\text{LT}_< g$ divides ϵ_i).

Definition 2 (Normal Form)**.** *Let f be a polynomial in $\mathbb{A}$. The normal form of f is defined w.r.t. a monomial ordering $<$ and denoted $NF_<(f)$: $NF_<(f)$ is the unique polynomial in $\mathbb{A}$ such that no term of $NF_<(f)$ is divisible by a leading term of a polynomial in $\mathbb{G}_>$ and there exist $g \in \mathcal{I}$ such that $f = g + NF_<(f)$. That is to say, $NF_<$ is a (linear) projection of $\mathbb{A}$ on $V_\mathcal{I}$. We recall that for any polynomials f, g, h we have $NF_<(fg) = NF_<(NF_<(f)g) = NF_<(NF_<(f)NF_<(g))$.*

Let ψ be the representation of $V_\mathcal{I}$ as a subspace of $\mathbb{K}^D$ associated to the canonical basis B:

$$\psi : \begin{pmatrix} V_\mathcal{I} & \to & \mathbb{K}^D \\ \sum_{i=1}^D \alpha_i \epsilon_i & \mapsto & [\alpha_1, \dots, \alpha_D]^t \, . \end{pmatrix}$$

We call *multiplication matrices*, denoted $T_1, \dots, T_n$, the matrix representation of the multiplication by $x_1, \dots, x_n$ in $V_\mathcal{I}$. That is to say, the i^{th} column of the matrix T_j is given by $\psi(\text{NF}_<(\epsilon_i x_j)) = [c_{i,1}^{(j)}, \dots, c_{i,D}^{(j)}]^t$ hence, $T_k = \left(c_{i,j}^{(k)}\right)_{i,j=1,\dots,D}$.

The LEX Gröbner basis of an ideal $\mathcal{I}$ has a triangular form. In particular, when $\mathcal{I}$ is zero-dimensional, its LEX Gröbner basis always contains a univariate polynomial. In general, the expected form of a LEX Gröbner basis is the *Shape Position*.

Definition 3 (Shape Position)**.** *An ideal of $\mathbb{A}$ is in shape position if its LEX Gröbner basis is of the form $\mathbb{G}_{lex} = \{x_1 - h_1(x_n), \dots, x_{n-1} - h_{n-1}(x_n), h_n(x_n)\}$ where $h_1, \dots, h_{n-1}$ are univariate polynomials of degree less than D and h_n is a univariate polynomial of degree D.*

When the field $\mathbb{K}$ is $\mathbb{Q}$ or when its characteristic p is sufficiently large, almost all zero-dimensional ideals have, up to a linear change of coordinates, a LEX Gröbner basis in *Shape Position* [28]. A characterization of the zero-dimensional ideals that can be placed in shape position has been given in [6]. A less general result [23, 29] usually called the *Shape Lemma* is the following: an ideal $\mathcal{I}$ is said to be radical if for any polynomial in $\mathbb{A}$, $f^k \in \mathcal{I}$ implies $f \in \mathcal{I}$. Up to a linear change of coordinates, any radical ideal has a LEX Gröbner basis in *Shape Position*. From now on, all the ideals considered in this paper will be zero-dimensional and will have a LEX Gröbner basis in *Shape Position*. Moreover, we fix the DRL order for the basis of $V_{\mathcal{I}}$ that is to say that $B = \{\epsilon_1, \ldots, \epsilon_D\}$ will always denotes the canonical vector basis of $V_{\mathcal{I}}$ w.r.t. the DRL order. Since for *Shape Position* ideals the LEX Gröbner basis is described by n univariate polynomials we will call it the "univariate polynomial representation" of the ideal or, up to multiplicities, of its variety of solutions.

In the following section, we present an algorithm to compute the LEX Gröbner basis of a *Shape Position* ideal. This algorithm assumes the DRL Gröbner basis and a multiplication matrix to be known. The computation of the multiplication matrices is treated in Section 4.

3 Univariate polynomial representation using structured linear algebra

In this section, we present an algorithm to compute univariate polynomial representation. This algorithm follows the one described in [21]. The main difference is that this new algorithm and its complexity study do not take into account any structure of the multiplication matrices (in particular any sparsity assumption).

Let $\mathbb{G}_{\text{lex}} = \{h_n(x_n), x_{n-1} - h_{n-1}(x_n), \ldots, x_1 - h_1(x_n)\}$ be the LEX Gröbner basis of $\mathcal{I}$. Given the multiplication matrices $T_1, \ldots, T_n$, an algorithm to compute the univariate polynomial representation has to find the n univariate polynomials $h_1, \ldots, h_n$. For this purpose, we can proceed in two steps. First, we will compute h_n. Then, by using linear algebra techniques, we will compute the others univariate polynomials $h_1, \ldots, h_{n-1}$.

Remark 1. *In this section, for simplicity, we present a probabilistic algorithm to compute the univariate polynomial representation. However, to obtain a deterministic algorithm it is sufficient to adapt the deterministic algorithm for radical ideals admitting a LEX Gröbner basis in Shape Position given in [20] in exactly the same way we adapt the probabilistic version.*

3.1 Computation of h_n

To compute h_n we have to compute the minimal polynomial of T_n. To this end, we use the first part of the Wiedemann probabilistic algorithm which succeeds with good probability if the field $\mathbb{K}$ is sufficiently large, see [43].

Let $\mathbf{r}$ be a random column vector in $\mathbb{K}^D$ and $\mathbf{1} = \psi(1)^t = [1, 0, \ldots, 0]^t$. If $a = [a_1, \ldots, a_D]$ and $b = [b_1, \ldots, b_D]$ are two vectors of $\mathbb{K}^D$, we denote by (a, b) the dot product of a and b defined by $(a, b) = \sum_{i=1}^{D} a_i b_i$. If $\mathbf{r}_1, \ldots, \mathbf{r}_k$ are column vectors then we denote by $(\mathbf{r}_1 | \ldots | \mathbf{r}_k)$ the matrix with D rows and k columns obtained by joining the vectors $\mathbf{r}_i$ vertically.

Let $S = [(\mathbf{r}, T_n^j \mathbf{1}) \mid j = 0, \ldots, 2D - 1]$ be a linearly recurrent sequence of size $2D$. By using for instance the Berlekamp-Massey algorithm [34], we can compute the minimal polynomial of S denoted μ. If $\deg(\mu(x_n)) = D$ then we deduce that $\mu(x_n) = h_n(x_n) \in \mathbb{G}_{\text{lex}}$ since μ is a divisor of f_n.

In order to compute efficiently S, we first notice that $(\mathbf{r}, T_n^j \mathbf{1}) = (T^j \mathbf{r}, \mathbf{1})$ where $T = T_n^t$ is the transpose matrix of T_n. Then, we compute $T^2, T^4, \ldots, T^{2^{\lceil \log_2 D \rceil}}$ using binary powering with $\lceil \log_2 D \rceil$ matrix multiplications. Similarly to [27], the vectors $T^j \mathbf{r}$ for $j = 0, \ldots, (2D - 1)$ are computed by induction in

$\log_2 D$ steps:

$$
\begin{array}{rcl}
T^2(T\mathbf{r} \mid \mathbf{r}) & = & (T^3\mathbf{r} \mid T^2\mathbf{r}) \\
T^4(T^3\mathbf{r} \mid T^2\mathbf{r} \mid T\mathbf{r} \mid \mathbf{r}) & = & (T^7\mathbf{r} \mid T^6\mathbf{r} \mid T^5\mathbf{r} \mid T^4\mathbf{r}) \\
 & \vdots & \\
T^{2^{\lceil \log_2(D) \rceil}}(T^{2^{\lceil \log_2(D) \rceil}-1}\mathbf{r} \mid \cdots \mid \mathbf{r}) & = & (T^{2D-1}\mathbf{r} \mid T^{2D-2}\mathbf{r} \mid \cdots \mid T^{2^{\lceil \log_2(D) \rceil}}\mathbf{r}) .
\end{array} \tag{3a}
$$

3.2 Recovering $h_1, \ldots, h_{n-1}$

We write $h_i = \sum_{k=0}^{D-1} \alpha_{i,k} x_n^k$ for $i = 1, \ldots, n-1$ where $\alpha_i \in \mathbb{K}$ are unknown. We have for $i = 1, \ldots, n-1$:

$$
x_i - h_i \in \mathbb{G}_{\text{lex}} \text{ is equivalent to } 0 = \mathrm{NF}_{\text{drl}}\left(x_i - \sum_{k=0}^{D-1} \alpha_{i,k} x_n^k\right) = T_i \mathbf{1} - \sum_{k=0}^{D-1} \alpha_{i,k} T_n^k \mathbf{1} .
$$

Multiplying the last equation by T_n^j for any $j = 0, \ldots, (D-1)$ and taking the scalar product we deduce that:

$$
0 = (\mathbf{r}, T_n^j(T_i\mathbf{1})) - \sum_{k=0}^{D-1} \alpha_{i,k} (\mathbf{r}, T_n^{k+j}\mathbf{1}) = (T^j\mathbf{r}, T_i\mathbf{1}) - \sum_{k=0}^{D-1} \alpha_{i,k} (T^{k+j}\mathbf{r}, \mathbf{1}) \tag{3b}
$$

Hence, we can recover h_i, for $i = 1, \ldots, n-1$ by solving $n-1$ structured linear systems:

$$
\underbrace{\begin{pmatrix} (T^0\mathbf{r}, T_i\mathbf{1}) \\ (T^1\mathbf{r}, T_i\mathbf{1}) \\ \vdots \\ (T^{D-1}\mathbf{r}, T_i\mathbf{1}) \end{pmatrix}}_{\mathbf{b_i}} = \underbrace{\begin{pmatrix} (T^0\mathbf{r}, \mathbf{1}) & (T^1\mathbf{r}, \mathbf{1}) & \ldots & (T^{D-1}\mathbf{r}, \mathbf{1}) \\ (T^1\mathbf{r}, \mathbf{1}) & (T^2\mathbf{r}, \mathbf{1}) & \ldots & (T^{D}\mathbf{r}, \mathbf{1}) \\ \vdots & \vdots & \ddots & \vdots \\ (T^{D-1}\mathbf{r}, \mathbf{1}) & (T^{D}\mathbf{r}, \mathbf{1}) & \ldots & (T^{2D-2}\mathbf{r}, \mathbf{1}) \end{pmatrix}}_{\mathcal{H}} \underbrace{\begin{pmatrix} c_{i,0} \\ c_{i,1} \\ \vdots \\ c_{i,D-1} \end{pmatrix}}_{\mathbf{c_i}} \tag{3c}
$$

Note that the linear system (3c) has a unique solution since from [25] the rank of $\mathcal{H}$ is given by the degree of the minimal polynomial of S which is exactly D in our case. The following lemma tell us that we can compute $T_i\mathbf{1}$ without knowing T_i.

Lemma 1. *The vectors $T_i\mathbf{1}$ for $i = 1, \ldots, n-1$ can be read from $\mathbb{G}_{drl}$.*

Proof. We have to consider the two cases $\mathrm{NF}_{\text{drl}}(x_i) \neq x_i$ or $\mathrm{NF}_{\text{drl}}(x_i) = x_i$.

First, if $\mathrm{NF}_{\text{drl}}(x_i) \neq x_i$ then there exists $g \in \mathbb{G}_{\text{drl}}$ such that $\mathrm{LT}_{\text{drl}}(g)$ divides x_i. This implies that g is a *linear* equation:

$$
x_i + \sum_{j>i}^{n} \alpha_{i,j} x_j + \alpha_{i,0} \text{ with } \alpha_{i,j} \in \mathbb{K} . \tag{3d}
$$

Hence, $\mathrm{NF}_{\text{drl}}(x_i) = -\sum_{j>i}^{n} \alpha_{i,j} x_j - \alpha_{i,0}$ and $T_i\mathbf{1} = -[\alpha_{i,0}, 0, \ldots, 0, \alpha_{i,i+1}, \ldots, \alpha_{i,n}, 0, \ldots]^t$. Otherwise $\mathrm{NF}_{\text{drl}}(x_i) = x_i$ so that $T_i\mathbf{1} = [0, \ldots, 0, 1, 0, \ldots, 0]^t$. $\square$

Hence, once the vectors $T^j\mathbf{r}$ have been computed for $j = 0, \ldots, (2D-1)$, we can deduce directly the Hankel matrix $\mathcal{H}$ with no computation but scalar products would seem to be needed to obtain the vectors $\mathbf{b}_i$. However, by removing the linear equations from $\mathbb{G}_{\text{drl}}$ we can deduce the $\mathbf{b}_i$ without arithmetic operations.

Linear equations in $\mathbb{G}_{\text{drl}}$. Let denote by $\mathbb{L}$ the set of polynomials in $\mathbb{G}_{\text{drl}}$ of total degree 1 (usually $\mathbb{L}$ is empty). We define $\mathcal{L} = \{j \in \{1, \dots, n-1\}$ such that $\text{NF}_{\text{drl}}(x_j) \neq x_j\}$ and $\mathcal{L}^c = \{1, \dots, n-1\} \backslash \mathcal{L}$ so that $\{x_i \mid i \in \mathcal{L}\} = \text{LT}_{\text{drl}}(\mathbb{L})$. In other words there is no linear form in $\mathbb{G}_{\text{drl}}$ with leading term x_i when $i \in \mathcal{L}^c$.

We first solve the linear systems (3c) for $i \in \mathcal{L}^c$: we know from the proof of Lemma 1 that $T_i\mathbf{1} = [0, \dots, 0, 1, 0, \dots, 0]^t$. Hence, the components $(T^j\mathbf{r}, T_i\mathbf{1})$ of the vector $\mathbf{b}_i$ can be extracted directly from the vector $T^j\mathbf{r}$. By solving the corresponding linear system we can recover $h_i(x_n)$ for all $i \in \mathcal{L}^c$.

Now we can easily recover the other univariate polynomials $h_i(x_n)$ for all $i \in \mathcal{L}$: by definition of $\mathcal{L}$ we have

$$l_i = x_i + \sum_{j \in \mathcal{L}^c} \alpha_{i,j}x_j + \alpha_{i,n}x_n + \alpha_{i,0} \in \mathbb{L} \subset \mathbb{G}_{\text{drl}} \text{ with } \alpha_{i,j} \in \mathbb{K}.$$

Hence the corresponding univariate polynomial $h_i(x_n)$ is simply computed by the formula: $h_i(x_n) = -\sum_{j \in \mathcal{L}^c} \alpha_{i,j}h_j(x_n) - \alpha_{i,n}h_n(x_n) - \alpha_{i,0}$. Hence we have reduced the number of linear systems (3c) to solve from $n-1$ to $n - \#\mathcal{L} - 1$.

We conclude this section by summarizing the algorithm to compute univariate polynomial representation in Algorithm 3. For a deterministic version of Algorithm 3, we refer the reader to Remark 1. In the next section, we discuss how to compute the multiplication matrices.

Algorithm 3: Univariate polynomial representation

Input : The multiplication matrix T_n and the DRL Gröbner basis $\mathbb{G}_{\text{drl}}$ of an ideal $\mathcal{I}$.
Output: Return the LEX Gröbner basis $\mathbb{G}_{\text{lex}}$ of $\mathcal{I}$ or *fail*.

Compute T^{2^i} for $i = 0, \dots, \log_2 D$ and compute $T^j\mathbf{r}$ for $j = 0, \dots, (2D-1)$ using induction (3a). Deduce the linearly recurrent sequence S and the Hankel matrix $\mathcal{H}$;
$h_n(x_n) :=$ BerlekampMassey(S) ;
if $\deg(h_n) = D$ **then**
  Let $\mathcal{L}^c = \{j \in \{1, \dots, n-1\}$ such that $\text{NF}_{\text{drl}}(x_j) = x_j\}$ and $\mathcal{L} = \{1, \dots, n-1\} \backslash \mathcal{L}^c$;
  for $j \in \mathcal{L}^c$ **do**
    Deduce $T_j\mathbf{1}$ and $\mathbf{b}_j$ then solve the structured linear system $\mathcal{H}\,\mathbf{c}_j = \mathbf{b}_j$;
    $h_j(x_n) := \sum_{i=0}^{D-1} c_{j,i}x_n^i$ where $c_{j,i}$ is the ith component of the vector $\mathbf{c}_j$;
  for $j \in \mathcal{L}$ **do**
    $h_j(x_n) := -\sum_{i \in \mathcal{L}^c} \alpha_{j,i}h_i(x_n) - \alpha_{j,n}h_n(x_n) - \alpha_{j,0}$ where $\alpha_{j,i}$ is the ith coefficient of the linear form whose leading term is x_j;
  return $[x_1 - h_1(x_n), \dots, x_{n-1} - h_{n-1}(x_n), h_n(x_n)]$;
else return *fail*;

---

4 Multiplication matrices

4.1 The original algorithm in $O(nD^3)$

To compute the multiplication matrices, we need to perform the computation of the normal forms of all monomials $\epsilon_i x_j$ where $1 \leq i \leq D$ and $1 \leq j \leq n$.

Proposition 1 ([19]). *Let $F = \{\epsilon_i x_j \mid 1 \leq i \leq D, 1 \leq j \leq n\} \setminus B$ be the frontier of the ideal. Let $t = \epsilon_i x_j \in F$ then*

I. either $t = LT_{drl}(g)$ for some $g \in \mathbb{G}_{drl}$ hence, $NF_{drl}(t) = t - g$;

II. or $t = x_k t'$ *with* $t' \in F$ *and* $\deg(t') < \deg(t)$. *Hence, if* $NF_{drl}(t') = \sum_{l=1}^{s} \alpha_l \epsilon_l$ *with* $\epsilon_s <_{drl} t'$, $NF_{drl}(t) = NF_{drl}(x_k NF_{drl}(t')) = \sum_{l=1}^{s} \alpha_l NF_{drl}(\epsilon_l x_k)$.

From this proposition, it is not difficult to see that the normal form of all the monomials $\epsilon_i x_j$ can be easily computed if we consider them in increasing order. Indeed, let $t = \epsilon_i x_j$ for some $i \in \{1, \dots, D\}$ and $j \in \{1, \dots, n\}$. Assume that we have already computed the normal form of all monomials less than t and of the form $\epsilon_{i'} x_{j'}$. If t is in B or is a leading term of a polynomial in $\mathbb{G}_{\mathrm{drl}}$ then its normal form is trivially known. If t is of type (II) of Proposition 1 then $t = x_k t'$ with $t' <_{\mathrm{drl}} t$ hence $\mathrm{NF}_{\mathrm{drl}}(t') = \sum_{i=l}^{s} \alpha_l \epsilon_l$ is known. Finally, $\mathrm{NF}_{\mathrm{drl}}(t) = \sum_{l=1}^{s} \alpha_l \mathrm{NF}_{\mathrm{drl}}(x_k \epsilon_l)$ with $x_k \epsilon_l <_{\mathrm{drl}} x_k t' = t$ for all $l = 1, \dots, s$. Thus the normal forms of $x_k \epsilon_l$ are known for all $l = 1, \dots, s$ and we can compute $\mathrm{NF}_{\mathrm{drl}}(t)$ in D^2 arithmetic operations. This yields the algorithm proposed in [19]. However, since the cardinal of the frontier F can be bounded by $n\,D$ the overall complexity is $O(nD^3)$ arithmetic operations.

4.2 Computing the multiplication matrices using fast linear algebra

Another way to compute the normal form of a term t is to find the unique polynomial in the ideal whose leading term is t and the others terms correspond to monomials in B. Hence, to compute the multiplication matrices, we look for the polynomial $t - \mathrm{NF}_{\mathrm{drl}}(t)$ for any t in the frontier F (see Proposition 1). Therefore, to compute these polynomials we proceed in two steps. First, we construct a polynomial in the ideal whose leading term is t. If t is the leading term of a polynomial g in $\mathbb{G}_{\mathrm{drl}}$ then the desired polynomial is g itself. Otherwise, t is of type II of Proposition 1 and $t = x_k t'$ with $t' \in F$ and $\deg(t') < \deg(t)$. We will proceed degree by degree so that we can assume we know a polynomial f' in the ideal whose leading term is t'; then the desired polynomial is $f = x_k f'$. Next, once we have all the polynomials f with all possible leading terms t of some degree d, we can recover the canonical form $t - \mathrm{NF}_{\mathrm{drl}}(t)$ by reducing f with respect to the others polynomials whose leading terms are less than t. By computing a reduced row echelon form of the Macaulay matrix (the matrix representation) of all these polynomials, we can reduced all of them simultaneously.

Following the idea presented above, we can now describe Algorithm 4 for computing all the multiplication matrices T_i. Assuming that F is sorted in increasing order w.r.t. $<_{drl}$, we define the linear map ϕ:

$$\phi : \left(\begin{array}{ccc} \mathbb{A} & \to & \mathbb{K}^{D+\#F} \\ \sum_{i=1}^{D} \alpha_i \epsilon_i + \sum_{j=1}^{\#F} \beta_j t_j & \mapsto & (\beta_{\#F}, \dots, \beta_1, \alpha_1, \dots, \alpha_D) \,. \end{array} \right)$$

Let M be a row indexed matrix by all the monomials in F. Let m be a monomial in F and i the position of m in F, $M[m]$ denotes the row of M of index m *i.e.* the $(\#F - i + 1)^{\mathrm{th}}$ row of M containing a polynomial of leading term m. If T is a matrix, $T[*, i]$ denotes the i^{th} column of T.

Proposition 2. *Algorithm 4 is correct.*

Proof. The key point of the algorithm is to ensure that for each monomial in F its normal form is computed and stored in NF before we use it. We will prove the following loop invariant for all d in $\{d_{\min}, \dots, d_{\max}\}$.

Loop invariant: at the end of step d, all the normal forms of the monomials of degree d in the frontier F *are computed and are stored in NF. Moreover, the* m^{th} *row of the matrix* M *contains* $\phi(m - NF_{drl}(m))$ *for any monomial* $m \in F_d$.

First, we assume that $d = d_{\min}$. Then, each monomial t of degree d in F is of type (I) of Proposition 1. Indeed, if t was of type (II) then there exists t' in F of degree $d - 1$ which divides t. This is impossible because $t' \in F_{d_{\min}-1} = \emptyset$. Hence, the normal form of t for $t \in F_{d_{\min}}$, is known and $M[t]$ contains $\phi(g)$ with g the unique element of $\mathbb{G}_{\mathrm{drl}}$ such that $\mathrm{LT}_{\mathrm{drl}}(g) = t$. Hence, $M[t] = \phi(g) = \phi(t - \mathrm{NF}_{\mathrm{drl}}(t))$. Moreover, since $\mathbb{G}_{\mathrm{drl}}$ is a reduced Gröbner basis , the matrix M is already in reduced row echelon form. Thus, the loop in Line 9 updates $\mathrm{NF}[t]$ for all $t \in F_d$.

Algorithm 4: Building multiplication matrices (in the following || does not mean parallel code but gives details about pseudo code on the left side).

Input : The DRL Gröbner basis $\mathbb{G}_{\text{drl}}$ of an ideal $\mathcal{I}$.
Output: The n multiplication matrices $T_1, \ldots, T_n$.

1 Compute $B = \{\epsilon_1 < \cdots < \epsilon_D\}$ and $F = \{x_i\epsilon_j \mid i = 1, \ldots, n \text{ and } j = 1, \ldots, D\} \setminus B$, $S := \#F$;
2 $d_{\min} := \min(\{\deg(t) \mid t \in F\})$; $d_{\max} := \max(\{\deg(t) \mid t \in F\})$; NF := [];
3 $M :=$ the zero matrix of size $nD \times (n+1)D$ row indexed by all the monomials in F;
4 **for** $d = d_{\min}$ **to** $d_{\max}$ **do**
5 $\quad F_d := Sort(\{t \in F \mid \deg(t) = d\}, <_{drl})$;
6 $\quad$ **for** $m \in F_d$ **do**
7 $\quad\quad$ Check if we can find: $(i) g \in \mathbb{G}_{\text{drl}}$ such that $\text{LT}_{\text{drl}}(g) = m$; (ii) $t' \in F$ such that $m = x_k t'$. Add the corresponding row to the matrix M; || **if** $m = LT_{drl}(g)$ **then** $M[m] := \phi(g)$; **else** Find x_k and $t' \in F_{d-1}$ such that $m = x_k t'$; $M[m] := \phi(m - x_k \text{NF}[t'])$;
8 $\quad M :=$ ReducedRowEchelonForm(M) ;
9 $\quad$ **for** $i = 1$ **to** s_d **do**
10 $\quad\quad$ Read $\text{NF}_{\text{drl}}(m)$ from M; || $\text{NF}[m] := -\sum_{j=1}^{D} M[m, S+j]\, \epsilon_j$;
11 Construct $T_1, \ldots, T_n$ from NF; || **for** ϵ *in* B **do** $\text{NF}[\epsilon] := \epsilon$; **for** t *in* $F \cup B$ **do** **for** x_i *s.t.* x_i *divides* t *and* $\frac{t}{x_i} = \epsilon_j \in B$ **do** $T_i[*, j] := \psi(\text{NF}[t])$;

return $T_1, \ldots, T_n$;

Let $d > d_{\min}$, we now assume that the loop invariant is true for any degree less than d. For all $t \in F_d$ the t^{th} row of M contains either $\phi(t - \text{NF}_{\text{drl}}(t))$ if t is of type (I) or $\phi(t - x_k\text{NF}[t'])$ if t is of type (II). Since $\deg(t') = d - 1$, by induction its normal form is known and in NF. Hence $\text{NF}[t'] = \text{NF}_{\text{drl}}(t')$ and $M[t] = \phi(x_k(t' - \text{NF}_{\text{drl}}(t')))$. A first consequence is that, before Line 8, since we sort F_d at each step, M is an upper triangular matrix with $M[t, t] = 1$ for all $t \in F_d$, see Figure 1. Note that sorting F_d is required only to obtain this triangular form. Let f be the polynomial $\text{NF}_{\text{drl}}(t')$. Writing $f = \sum_{j=1}^{D} \lambda_j \epsilon_j$ we have that $\lambda_j = 0$ if $\deg(\epsilon_j) \geq d$ since $\deg(\text{NF}_{\text{drl}}(t')) \leq \deg(t') = d - 1$. So that $f = \sum_{j=1}^{k} \lambda_j \epsilon_j$ such that $\deg(\epsilon_j) < d$ when $j \leq k$. Now for all j such that $1 \leq j \leq k$ we are in one of the following cases:

1. $x_k\epsilon_k \in B$ so that $\text{NF}_{\text{drl}}(x_k\epsilon_k) = x_k\epsilon_k$ is already reduced.
2. $x_k\epsilon_k \in F$. Since $d' = \deg(x_k\epsilon_k) \leq d$ it implies that $x_k\epsilon_k \in F_{d'}$ so that the row $M[x_k\epsilon_k]$ has been added to M.

Moreover, since each row of the matrix M contains polynomial in the ideal $\langle \mathbb{G}_{\text{drl}} \rangle$ after the computation of the row echelon form, the rows of the matrix M contain also polynomials in $\langle \mathbb{G}_{\text{drl}} \rangle$ being linear combination of the previous polynomials. Hence, after the computation of the row echelon form of M, the row $M[t]$ is equal to $\phi(t - \text{NF}_{\text{drl}}(t))$.

By induction, this finishes the proof of the loop invariant and then of the correctness of Algorithm 4. $\square$

5 Polynomial equations with fixed degree: the tame case

The purpose of this section, is to analyze the asymptotic complexity of Algorithm 3 and Algorithm 4 when the degrees of the equations of the input system are uniformly bounded by a fixed integer $d > 1$ and to establish the first main result of this paper.

5.1 General Complexity analysis

We first analyse Algorithm 3 to compute the univariate polynomial representation given the last multiplication matrix.

Proposition 3. *Given the multiplication matrix T_n and the DRL Gröbner basis $\mathbb{G}_{drl}$ of an ideal in Shape Position, its LEX Gröbner basis can be probabilistically computed in $O(\log_2(D)(D^\omega+n\log_2(D)D))$ where D is the number of solutions. Expressed with the input parameters of the system to solve the complexity is $O(nd^{\omega n})$ where $d>1$ is a (fixed) bound on the degree of the input polynomials.*

Proof. As usual $T=T_n^t$ is the transpose matrix of T_n. Using the induction (3a), the vectors $T^j\mathbf{r}$ can be computed for all $j=0,\ldots,(2D-1)$ in $O(\log_2(D)D^\omega)$ field operations. Then the linear recurrent sequence S and the matrix $\mathcal{H}$ can be deduced with no cost. The Berlekamp-Massey algorithm compute the minimal polynomial of S in $O(D\log_2^2(D))$ field operations [8,25].

As defined in Section 3.2, $\mathcal{L}=\{j\in\{1,\ldots,n-1\}$ such that $\mathrm{NF}_{\mathrm{drl}}\,(x_j)\neq\ x_j\}$ and $\mathcal{L}^c=\{1,\ldots,n-1\}\backslash\mathcal{L}$. The right hand sides of the linear systems $\mathbf{b}_i$ can be computed without field operations when $i\in\mathcal{L}^c$. Since the matrix $\mathcal{H}$ is a non singular Hankel matrix, the $\#\mathcal{L}^c$ linear systems (3c) can be solved in $O(\#\mathcal{L}^c\log_2^2(D)D)=O(n\log_2^2(D)D)$ field operations. Then, to recover all the $h_i(x_n)$ for $i\in\mathcal{L}$ we perform $O(\#\mathcal{L}\#\mathcal{L}^cD)=O(n^2D)$ multiplications and additions in $\mathbb{K}$.

Since the Bézout's bound allows to bound D by d^n with d a fixed integer we have $\log_2(D)\leq n\log_2(d)$ and the arithmetic complexity of Algorithm 3 is $O(\log_2(D)(D^\omega+n\log_2(D)D))$ which can be expressed in terms of d and n as $O(nd^{\omega n})$. ☐

Note that the deterministic version, mentioned in Remark 1 have a complexity in $O(\log_2(D)D^\omega+D^2(n+\log_2(D)\log_2(\log_2(D))))$ arithmetic operations, thanks to induction (3a) and section 3.2.2 in [20]. This deterministic version computes the LEX Gröbner basis of the radical of the ideal in input when the ideal is in *Shape Position*,. In our case, this is not restricting since in Problem 1 we assume that all the roots of the system are simple which is equivalent to say that the ideal generated by the polynomial is radical.

Proposition 4. *Let T_n be the multiplication matrix and $\mathbb{G}_{drl}$ be the DRL Gröbner basis of a radical ideal $\mathcal{I}$ in Shape Position. There is a deterministic algorithm which computes the LEX Gröbner basis of $\mathcal{I}$ in $O(\log_2(D)D^\omega+D^2(n+\log_2(D)\log_2(\log_2(D))))$ (or in $O(nd^{\omega n})$) arithmetic operations in $\mathbb{K}$.*

Now, to complete the first algorithm, we deal with the complexity of Algorithm 4 to compute the multiplication matrices. Note that in proposition 3 and 4 only the last matrix T_n is needed. Before to consider the complexity of Algorithm 4, we first discuss about the complexity of computing B and F.

Lemma 2. *Given $\mathbb{G}_{drl}$ (resp. B) the construction of B (resp. F) requires at most $O(n^3D^2)$ (resp. $O(nD^2+n^2D)$) elementary operations which can be decreased to $O(nD)$ (resp. $O(n^2D)$) elementary operations if a hash table is used.*

Proof. It is well known that the canonical basis B can be computed in polynomial time (but no arithmetic operations). Nevertheless, in order to be self contained we describe an elementary algorithm to compute B. We start with the monomial 1 and we multiply it by all the variables x_i which gives n new monomials to consider. If the new monomials are not divisible by a leading term of a polynomial in $\mathbb{G}_{\mathrm{drl}}$ then we keep it otherwise we discard it. At each step we multiply by the variables x_i only the monomials of highest degree that we have kept and we proceed until the step where all the new monomials are discarded. Hence, we have to test the irreducibility of all the elements in $F\cup B$ whose total number is bounded by $(n+1)D$. Since $\mathrm{LT}_{\mathrm{drl}}\,(\mathbb{G}_{\mathrm{drl}})\subset F$ we can bound the number of elements of $\mathbb{G}_{\mathrm{drl}}$ by nD. Therefore, to compute B we have to test the divisibility of $(n+1)D$ monomials by at most nD monomials. Hence, the construction of B can be done in $O(n^3D^2)$ elementary operations. Note that by using a hash table and assuming we have no memory limit, for each monomial we can test its divisibility by a leading term of polynomials in $\mathbb{G}_{\text{drl}}$ in $O(1)$ operations. In that case B can be constructed in $O(nD)$ elementary operations.

From B, the construction of F requires nD monomials multiplications *i.e.* n^2D additions of integers. Moreover, removing B of F can be done by testing if $(n+1)D$ monomials are in B in at most $O(nD^2)$ elementary operations which can be decreased to $O(nD)$ if we use a hash table. $\square$

Now we seen how constructing B and F, the complexity of Algorithm 4 is treated in the following proposition.

Proposition 5. *Given the DRL Gröbner basis $\mathbb{G}_{drl}$ of an ideal, we can compute all the multiplication matrices in $O((d_{\max}-d_{\min})n^\omega D^\omega)$ (or in $O((d_{\max}-d_{\min})n^\omega d^{\omega n})$) arithmetic operations in $\mathbb{K}$ where $d_{\max}$ (resp. $d_{\min}$) is the maximal (resp. the minimal) degree of all the polynomials in $\mathbb{G}_{drl}$.*

Proof. Algorithm 4, computes all the multiplication matrices incrementally degree by degree. The frontier F can be written as the union of disjoint sets $F_\delta = \{t \in F \mid \deg(t) = \delta\}$ so that we define $s_\delta := \#F_\delta$ and $S_\delta := s_{d_{\min}} + \cdots + s_\delta$. The cost of the loop at Line 4 is, at each step, given by the complexity of computing the reduced row echelon form of M. In degree δ the shape of the matrix M is depicted on Figure 1 where $\mathbf{Id}(S_{\delta-1})$ is the $S_{\delta-1}\times S_{\delta-1}$ identity matrix, $\mathbf{0}(S_{\delta-1})$ is the $S_{\delta-1}\times s_\delta$ zero matrix, T is a $s_\delta \times s_\delta$ upper triangular matrix and B, C, D are dense matrices of respective size $s_\delta \times S_{\delta-1}$, $s_\delta \times D$, $S_{\delta-1}\times D$.

$$
M = \begin{array}{|cccc|ccc|ccc|}
\multicolumn{4}{c}{t \in F_\delta} & \multicolumn{3}{c}{t \in F_{\delta-1}\cup\cdots\cup F_{d_{\min}}} & \multicolumn{3}{c}{t\in B} \\
\hline
1 & \star & \cdots & \star & \star & \cdots & \star & \star & \cdots & \star \\
0 & 1 & \cdots & \star & \star & \cdots & \star & \star & \cdots & \star \\
\vdots & \mathbf{T} & \ddots & \vdots & \vdots & \mathbf{B} & \vdots & \vdots & \mathbf{C} & \vdots \\
0 & 0 & \cdots & 1 & \star & \cdots & \star & \star & \cdots & \star \\
\hline
0 & 0 & \cdots & 0 & 1 & \cdots & 0 & \star & \cdots & \star \\
\vdots & \mathbf{0}(S_{\delta-1},s_\delta) & & \vdots & \mathbf{Id}(S_{\delta-1}) & \ddots & & \vdots & \mathbf{D} & \vdots \\
0 & 0 & \cdots & 0 & 0 & \cdots & 1 & \star & \cdots & \star \\
\hline
\end{array}
$$

Figure 1: Shape of the matrix M of Algorithm 4.

Consequently the reduced row echelon form of M can be obtained from the following formula:

$$\text{ReducedRowEchelonForm}(M) = \left[\begin{array}{c|c} \mathbf{Id}(S_\delta) & \begin{array}{c} T^{-1}(C-BD) \\ \hdashline D \end{array} \end{array}\right].$$

Since $s_\delta \leq S_\delta \leq S_{d_{\max}} \leq nD$ we can bound the complexity of computing the reduced row echelon form of M by $O(n^\omega D^\omega)$. From Lemma 2, the costs of the construction of B and F are negligible in comparison to the cost of loop in Line 4 which therefore gives the complexity of Algorithm 4: $O((d_{\max}-d_{\min})n^\omega D^\omega))$ arithmetic operations. Since $D \leq d^n$, this complexity can be written as $O((d_{\max}-d_{\min})n^\omega d^{\omega n})$. $\square$

5.2 Complexity for regular systems

Regular systems form an important family of polynomial systems. Actually, the complexity of computing a Gröbner basis of a regular system is well understood. Since the property of being regular is a generic property this also the typical behavior of polynomial systems.

Definition 4. *A sequence of non zero homogeneous polynomials $(f_1,\ldots,f_m) \in \mathbb{A}^m$ is regular if for all $i = 1,\ldots,m-1$, f_{i+1} does not divide 0 in $\mathbb{A}/\langle f_1,\ldots,f_i\rangle$. A sequence of non zero affine polynomials is regular if the sequence $(f_1^h,\ldots,f_m^h)$ is regular where f_i^h is the homogeneous part of highest degree of f_i.*

For regular systems we can bound accurately the values of $d_{\max}$ which is the maximal degree of $\mathbb{G}_{\text{drl}}$ and we can proof the first main result of this paper.

Theorem 5.1. *Let $\mathcal{S} = \{f_1, \dots, f_n\}$ be a polynomial system generating a radical ideal admitting a LEX Gröbner basis in Shape Position. Assume that $(f_1, \dots, f_n)$ is a regular sequence of polynomials whose degrees are uniformly bound by a fixed integer d i.e. $\deg(f_i) \le d$ for $i = 1, \dots, n$. The univariate polynomial representation of all the solutions of $\mathcal{S}$ can be computed using a deterministic algorithm in $O(d^{\omega n} + (dn^{\omega+1} + \log_2(D))D^{\omega})$ arithmetic operations in $\mathbb{K}$.*

Proof. For regular systems $d_{\max}$ can be bounded by the Macaulay bound [1,31]: $d_{\max} \le \sum_{i=1}^{n}(\deg(f_i) - 1) + 1 \le n(d-1) + 1$. Given the system $\mathcal{S}$ the complexity of computing the DRL Gröbner basis of $\langle S \rangle$ is bounded by [1]:

$$O\left(\binom{n + d_{\max}}{n}^{\omega}\right) = O\left(\binom{nd+1}{n}^{\omega}\right) = O(d^{\omega n})$$

arithmetic operations.

From this DRL Gröbner basis, according to Proposition 5, the multiplication matrix T_n can be computed in $O(dn^{\omega+1}D^{\omega})$ arithmetic operations.

Finally, from T_n and the DRL Gröbner basis, thanks to Proposition 4 the univariate polynomial representation can be computed by a deterministic algorithm in $O(\log_2(D)D^{\omega} + D^2(n + \log_2(D)\log_2(\log_2(D))))$ arithmetic operations. Since, F_4 [16], F_5 [17] and Algorithm 4 are deterministic algorithms this finishes the proof. □

Among regular systems, there are generic systems. A generic systems is a sequence of dense polynomials whose coefficients are unknowns or any random instantiations of these coefficients. Let $d_i = \deg(f_i)$ for all $i = 1, \dots, n$. Since the Bézout's bound allows to bound the number of solutions D by $\prod_{i=1}^{n} d_i \le d^n$ and since this bound is generically reached, we have generically that $D = \prod_{i=1}^{n} d_i \le d^n$ and we get the following corollary.

Corollary 1. *Let $\mathbb{K}$ be the rational field $\mathbb{Q}$ or a finite field $\mathbb{F}_q$. Let $\mathcal{S} = \{f_1, \dots, f_n\} \subset \mathbb{K}[x_1, \dots, x_n]$ be a generic polynomial system generating an ideal $\mathcal{I} = \langle \mathcal{S} \rangle$ of degree D. If $\mathcal{I}$ admits a LEX Gröbner basis in Shape Position and if the degree of each polynomial in $\mathcal{S}$ is uniformly bounded by a fixed integer d then there exists a deterministic algorithm which computes the univariate polynomial representation of the roots of $\mathcal{S}$ in $\widetilde{O}(D^{\omega})$ arithmetic operations where the notation $\widetilde{O}$ means that we neglect logarithmic factors in D and polynomial factors in n.*

In the next section, we study a first step towards the generalization of Theorem 5.1 to polynomial systems with equations of non fixed degree. More precisely, we are going to discuss what happens if one polynomial have a non fixed degree *i.e.* its degree depends on a parameter (for instance the number of variables). In this case, Theorem 5.1 does not apply but we present other arguments in order to obtain a similar complexity results for computing $\mathbb{G}_{\text{lex}}$ given $\mathbb{G}_{\text{drl}}$ and new ideas for its generalization.

6 A worst case ultimately not so bad

We consider, for instance, the following pathological case: $\deg(h_1) = \cdots = \deg(h_{n-1}) = 2$ and $\deg(h_n) = 2^n$. Then, $D = 2^{2n-1}$, $d_{\min} = 2$ and $d_{\max} = 2^n + n - 1$. In this context, the complexity of computing $\mathbb{G}_{\text{lex}}$ given $\mathbb{G}_{\text{drl}}$ seems to be in $O(\log_2^{\omega}(D)D^{\omega+\frac{1}{2}})$ arithmetic operations. However, we will show that an adaptation of Algorithm 4 allows to decrease this complexity.

In [35], Moreno-Socias studied the basis of the residue class ring $\mathbb{A}/\mathcal{I}$, w.r.t. the DRL ordering, for generic ideals. In particular, he shows that when the smallest variable x_n is in abscissa any section of the stairs of $\mathcal{I}$ has steps of height one and of depth two. That is to say, for any variable x_i with $i < n$ and for all instantiations of the others variables ($\{x_1, \ldots, x_{n-1}\} \setminus \{x_i\}$) the associated section of the stairs of $\mathcal{I}$ has the shape in Figure 2.

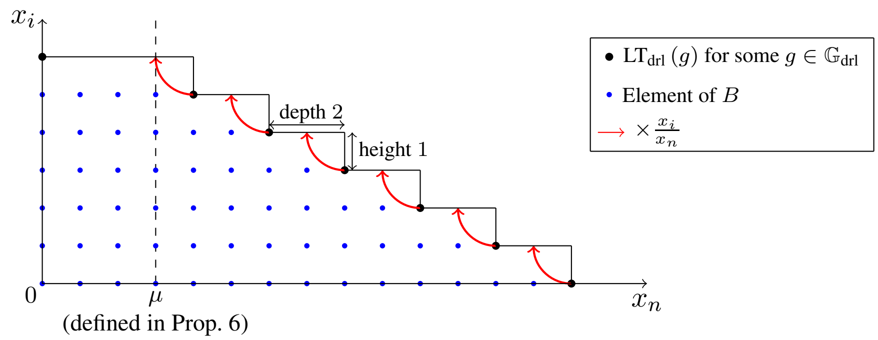

Figure 2: Section of the stairs of generic ideals with $\deg(x_j)$ fixed for all $j \in \{1, \ldots, n-1\} \setminus i$.

This shape is summarized in Proposition 6.

Proposition 6 (Moreno-Socias [35])**.** *Let* $\widetilde{B}_i = \{m = x^{\alpha_1} \cdots x_{n-1}^{\alpha_{n-1}} \mid m x_n^i \in B\}$. *Let* $\delta = \sum_{i=1}^{n}(\deg(h_i) - 1)$, $\delta^* = \sum_{i=1}^{n-1}(\deg(h_i) - 1)$ *and* $\sigma = \min\left(\delta^*, \lfloor \frac{\delta}{2} \rfloor\right)$. *Let* $\mu = \delta - 2\sigma$, *then*

a. $\widetilde{B}_0 = \cdots = \widetilde{B}_\mu$ *and* $\widetilde{B}_i = \widetilde{B}_{i+1}$ *for* $\mu < i < \delta$ *and* $i \not\equiv \delta \mod 2$*;*

b. The leading term of polynomials in $\mathbb{G}_{drl}$ *of degree* 0 *in* x_n *have degree at most* $\sigma + 1 = \bar{\sigma}$*;*

c. The leading term of polynomials in $\mathbb{G}_{drl}$ *of degree* α *in* x_n *with* $\mu < \alpha \leq \delta + 1$ *with* $\alpha \not\equiv \delta \mod 2$ *are all of total degree* $d + \alpha$ *where* $d = \max(\deg(m) \mid m \in \widetilde{B}_{\alpha-1})$*. Moreover, all these leading terms are exactly given by* $t = m x_n^\alpha$ *for all* $m \in \widetilde{B}_{\alpha-1}$ *of degree* d*;*

d. There is no leading term of polynomials in $\mathbb{G}_{drl}$ *of degree* $1, \ldots, \mu$ *in* x_n *or of degree* α *in* x_n *with* $\alpha > \delta + 1$ *or* $\mu \leq \alpha \leq \delta$ *and* $\alpha \equiv \delta \mod 2$.

In our case, we have $d_{\max} = \delta + 1$, $\delta^* = n - 1$, $\delta = 2^n + n - 2$, $\sigma = n - 1$ and $\mu = 2^n - n$. We can note that in this particular case, μ is very large which implies that a large part of the monomials of the form $\epsilon_i x_j$ are actually in B. We will show that in Algorithm 4 instead of computing the loop in Line 4 for $d = d_{\min}, \ldots, d_{\max}$ we can perform it only on restricted subset $d = d_{\min}, \ldots, \sigma(n-1)+1, \mu+1, \ldots, d_{\max}$. By consequence, the complexity of computing $\mathbb{G}_{\mathrm{lex}}$ given $\mathbb{G}_{\mathrm{drl}}$ will be in $O((d_{\max} - \mu + \sigma(n-1) - d_{\min}) n^\omega D^\omega) = O(\log_2^{\omega+2}(D) D^\omega)$ with $d_{\max} - \mu + \sigma(n-1) - d_{\min} = n^2 - 2 \sim \log_2^2(D)$.

Lemma 3. *Given the normal form of all monomials in* F *of degree less or equal to* $\sigma(n-1) + 1$ *we can compute all the normal forms of all monomials in* F *of degree less or equal than* μ *in less than* $O(nD^2)$ *arithmetic operations.*

Suppose that we know the normal form of the monomials of the forms $\epsilon_i x_j$ of degree less than μ which are not divisible by x_n. From these normal forms, the idea of the proof is to show that the normal form of all the monomials of the form $\epsilon_i x_j$ of degree less than μ and of degree $\alpha_n > 0$ in x_n is given by $x_n^{\alpha_n} \mathrm{NF}_{\mathrm{drl}}(t)$ where $\mathrm{NF}_{\mathrm{drl}}(t)$ is assumed to be known.

Proof. Let $t \in F$ of degree less or equal to μ. First, assume that x_n does not divide t. As $\mathcal{I}$ is zero dimensional, there exists $\eta_1, \ldots, \eta_{n-1} \in \mathbb{N}$ such that $x_i^{\eta_i}$ is a leading term of a polynomial in $\mathbb{G}_{\mathrm{drl}}$. Moreover, from Proposition 6, $\eta_i \leq \bar{\sigma}$. Hence, for all $\epsilon \in \widetilde{B}_0$, $\deg(\epsilon) \leq \sigma(n-1)$. The monomials in F not divisible by x_n are all of the form $x_i\epsilon$ with $i = 1, \ldots, n-1$ and $\epsilon \in \widetilde{B}_0$. Thus $\deg(t) \leq \sigma(n-1)+1$ and by hypothesis, its normal form is known.

Suppose now that x_n divides t and t is of type II of Proposition 1. We can write $t = x_n^{\alpha} t'$ where $\alpha \in \mathbb{N}^*$ such that $x_n \nmid t'$. From Proposition 6 item (d), t' is a leading term of a polynomial in $\langle \mathbb{G}_{\text{drl}} \rangle$. Moreover, $t \in F$ so $t = x_i \epsilon$ with $\epsilon \in B$. Suppose that $i = n$ hence, $\frac{t}{x_n} = \epsilon = x_n^{\alpha-1} t' \in \langle \mathbb{G}_{\text{drl}} \rangle$ which is impossible. Thus, $i \neq n$ and we have, $t' = \frac{t}{x_n^{\alpha}} = x_i \epsilon' \in F$ with $\epsilon' = \frac{\epsilon}{x_n^{\alpha}} \in B$. Therefore, from the first part of this proof, $\text{NF}_{\text{drl}}(t') = \sum_{i=1}^{s} \alpha_i \epsilon_i$, $\alpha_i \in \mathbb{K}$ is known. Finally, $\text{NF}_{\text{drl}}(t) = \sum_{i=1}^{s} \alpha_i \text{NF}_{\text{drl}}(x_n^{\alpha} \epsilon_i)$ with $\deg(x_n^{\alpha}\epsilon_i) \leq \mu$. Let k_i be such that $x_n^{k_i} | \epsilon_i$ and $x_n^{k_i+1} \nmid \epsilon_i$ as $\widetilde{B}_{k_i} = \widetilde{B}_{k_i+\alpha}$ then $x_n^{\alpha}\epsilon_i \in B$ and $\text{NF}_{\text{drl}}(t) = \sum_{i=1}^{s} \alpha_i x_n^{\alpha} \epsilon_i$.

By consequence, computing the normal form of t can be done in less than D arithmetic operations. As usual, we can bound the size of F by nD which finishes the proof. □

One can notice that Algorithm 3 – which computes univariate polynomial representation – takes as input only the multiplication matrix by the smallest variable. Thus in the proof of Theorem 5.1 we did not fully take advantage of this particularity. Hence, the next section is devoted to study if this matrix can be computed more efficiently than computing all the multiplication matrices. By studying the structure of the basis of the $\mathbb{K}$-vector space $\mathbb{A}/\mathcal{I}$ we will show that, up to a linear change of variables, T_n can be deduced from $\mathbb{G}_{\text{drl}}$. In the previous results, the algorithm restricting the order of magnitude of the degrees of the equations is Algorithm 4 to compute the multiplication matrices. Since, we need only T_n which can be computed very efficiently, the impact of such a result is that there exists a Las Vegas algorithm extending the result of Theorem 5.1 to polynomial systems whose equations have non fixed degree.

7 Polynomial equations with non-fixed degree: the wild case

In this section, in order to obtain our main result, we consider *initial* and *generic* ideals. The initial ideal of $\mathcal{I}$, denoted $\text{in}_<(\mathcal{I})$, is defined by $\text{in}_<(\mathcal{I}) = \{\text{LT}_<(f) \mid f \in \mathcal{I}\}$. A minimal set of generators of $\text{in}_<(\mathcal{I})$ is denoted $E(\mathcal{I})$, and is given by the leading terms of the polynomials in the Gröbner basis of $\mathcal{I}$ w.r.t. the monomial ordering $<$. To compute the multiplication matrix T_n we need to compute the normal forms of all monomials $\epsilon_i x_n$ for $i = 1, \ldots, D$ with $\epsilon_i \in B$. As mentioned in Section 4 a monomial of the form $\epsilon_i x_n$ can be either in B or in $E(\mathcal{I})$ or in $\text{in}_<(\mathcal{I}) \setminus E(\mathcal{I})$. As previously shown, the difficulty to compute T_n lies in the computation of the normal forms of monomials $\epsilon_i x_n$ that are in $\text{in}_<(\mathcal{I}) \setminus E(\mathcal{I})$). In this section, thanks to the study of the stairs, *i.e.* B, of generic ideals by Moreno-Socias, see Section 6, we first show that for generic ideals, *i.e.* ideals generated by generic systems (as defined in Section 5.2), all monomials of the form $\epsilon_i x_n$ are in B or in $E(\mathcal{I})$). Hence, the multiplication matrix T_n can be computed very efficiently. Then, we show that, up to a linear change of variables, this result can be extended to any ideal. According to these results, we finally propose an algorithm for solving the PoSSo problem whose complexity allows to obtain the second main result of this paper.

7.1 Reading directly T_n from the Gröbner basis

In the sequel, the arithmetic operations will be the addition or the multiplication of two operands in $\mathbb{K}$ that are different from ± 1 and 0. In particular we do not consider the change of sign as an arithmetic operation.

Proposition 7. *Let $\mathcal{I}$ be a generic ideal. Let t be a monomial in $E(\mathcal{I})$ i.e. a leading term of a polynomial in the DRL Gröbner basis of $\mathcal{I}$. If x_n divides t then for all $k \in \{1, \ldots, n-1\}$, $\frac{x_k t}{x_n} \in in_{drl}(\mathcal{I})$.*

Proof. This result is deduced from the shape of the stairs of $\mathcal{I}$ (see Figure 2 for a representation in dimension 2). Let $t = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ be a leading term of a polynomial in $\mathbb{G}_{\text{drl}}$ divisible by x_n *i.e.* $\alpha_n > 0$ and $m = x_1^{\alpha_1} \cdots x_{n-1}^{\alpha_{n-1}}$. We use the same notations as in Proposition 6.

From Proposition 6 item (d), since $t \in E(\mathcal{I})$ and $\alpha_n > 0$ we have $\alpha_n > \mu$ and $\alpha_n \not\equiv \delta \mod 2$. Then, from Proposition 6 item (c), $\deg(m)$ is the maximal degree reached by the monomials in $\widetilde{B}_{\alpha_{n-1}}$. Thus $x_k m \notin \widetilde{B}_{\alpha_{n-1}}$ for all $k \in \{1, \ldots, n-1\}$. As a consequence, for all $k \in \{1, \ldots, n-1\}$ we have $\frac{x_k t}{x_n} \in \text{in}_{drl}(\mathcal{I})$. $\square$

Consequently, from the previous proposition, we obtain the following result.

Theorem 7.1. *Given $\mathbb{G}_{drl}$ the DRL Gröbner basis of a generic ideal $\mathcal{I}$, the multiplication matrix T_n can be read from $\mathbb{G}_{drl}$ with no arithmetic operation.*

Proof. Suppose that there exists $i \in \{1, \ldots, D\}$ such that $t = x_n \epsilon_i$ is of type (II). Hence, $t = m\,\text{LT}_{\text{drl}}(g)$ for some $g \in \mathbb{G}_{\text{drl}}$ and $\deg(m) > 1$ with $x_n \nmid m$ (otherwise $\epsilon_i \notin B$). Then, there exists $k \in \{1, \ldots, n-1\}$ such that $x_k \mid m$. By consequence, from Proposition 7, we have $\epsilon_i = \frac{m}{x_k} \cdot \frac{x_k \text{LT}_{\text{drl}}(g)}{x_n} \in \text{in}_{drl}(\mathcal{I})$ which yields a contradiction. Thus, all monomials $t = x_n \epsilon_i$ are either in B or in $E(\mathcal{I})$ and their normal forms are known and given either by t (if $t \in B$) or by changing the sign of some polynomial $g \in \mathbb{G}_{\text{drl}}$ and removing its leading term. Note that by using a linked list representation (for instance), removing the leading term of a polynomial does not require arithmetic operation. $\square$

Thanks to the previous theorem, Algorithm 3 can be used to compute the LEX Gröbner basis of a generic ideal:

Corollary 2. *Let $\mathcal{I}$ be a generic ideal in Shape Position. From the DRL Gröbner basis$\mathbb{G}_{drl}$ of $\mathcal{I}$, its LEX Gröbner basis $\mathbb{G}_{lex}$ can be computed in $O(\log_2(D)(D^\omega + n\log_2(D)D))$ arithmetic operations with a probabilistic algorithm or $O(\log_2(D)D^\omega + D^2(n + \log_2(D)\log_2(\log_2(D))))$ arithmetic operations with a deterministic algorithm.*

However, polynomial systems coming from applications are usually not generic. Nevertheless, this difficulty can be bypassed by applying a linear change of variables. Let $g \in \text{GL}(\mathbb{K}, n)$ the ideal $g \cdot \mathcal{I}$ is defined as follows $g \cdot \mathcal{I} = \{f(g \cdot X) \mid f \in \mathcal{I}\}$ where X is the vector $[x_1, \ldots, x_n]$. By studying the structure of the *generic initial ideal* of $\mathcal{I}$ – that is to say, the initial ideal of $g \cdot \mathcal{I}$ for a generic choice of g – we will show that results of Proposition 7 and Theorem 7.1 can be generalized to non generic ideals, up to a random linear change of variables. Indeed, in [22] Galligo shows that for the characteristic zero fields, the generic initial ideal of any ideal satisfies a more general property than Proposition 7. Later, Pardue [38] extends this result to the fields of positive characteristic.

Definition 5. *Let $\mathbb{K}$ be an infinite field and $\mathcal{I}$ be an homogeneous ideal of $\mathbb{K}[x_1, \ldots, x_n]$. There exists a Zariski open set $U \subset GL(\mathbb{K}, n)$ and a monomial ideal $\mathcal{J}$ such that $\text{in}_{drl}(g \cdot \mathcal{I}) = \mathcal{J}$ for all $g \in U$. The generic initial ideal of $\mathcal{I}$ is denoted Gin$(\mathcal{I})$ and is defined by $\mathcal{J}$.*

The next result, is a direct consequence of [5, 22, 38] and summarized in [15, p.351–358]. This result allows to extend, up to a linear change of variables, Proposition 7 to non generic ideals.

Theorem 7.2. *Let $\mathbb{K}$ be an infinite field of characteristic $p \geq 0$. Let $\mathcal{I}$ be an homogeneous ideal of $\mathbb{K}[x_1, \ldots, x_n]$ and $\mathcal{J} = \text{Gin}(\mathcal{I})$. For the DRL ordering, for all generators m of $\mathcal{J}$, if x_i^t divides m and x_i^{t+1} does not divide m then for all $j < i$, the monomial $\frac{x_j}{x_i} m$ is in $\mathcal{J}$ if $t \not\equiv 0 \mod p$.*

Let $f = \sum_{i=0}^{d} f_i$ be an affine polynomial of degree d of $\mathbb{A}$ where f_i is an homogeneous polynomial of degree i. The homogeneous component of highest degree of f, denoted f^h, is the homogeneous polynomial f_d. Let $\mathcal{I}$ be an affine ideal *i.e.* generated by a sequence of affine polynomials. In the next proposition we highlight an homogeneous ideal having the same initial ideal than $\mathcal{I}$. This allows to extend the result of Theorem 7.2 to affine ideals.

Proposition 8. *Let $\mathcal{I} = \langle f_1, \ldots, f_s \rangle$ be an affine ideal. If $(f_1, \ldots, f_s)$ is a regular sequence, then there exists a Zariski open set $U_a \subset GL(\mathbb{K}, n)$ such that for all $g \in U_a$, $E(g \cdot \mathcal{I}) = E\left(Gin\left(\mathcal{I}^h\right)\right)$.*

Proof. Let f be a polynomial. We denote by f^h the homogeneous component of highest degree of f and $f^a = f - f^h$. Let $t \in \mathrm{in}_{drl}(\mathcal{I})$, there exists $f \in \mathcal{I}$ such that $\mathrm{LT}_{\mathrm{drl}}(f) = t$. Since, $f \in \mathcal{I}$ and $(f_1^h, \ldots, f_s^h)$ is assumed to be a regular sequence then there exist $h_1, \ldots, h_s \in \mathbb{K}[x_1, \ldots, x_n]$ such that $f = \sum_{i=1}^s h_i f_i = \sum_{i=1}^s h_i f_i^h + \sum_{i=1}^s h_i f_i^a$ with $\deg(h_i f_i) \leq \deg(f)$ for all $i \in \{1, \ldots, s\}$ and there exists $j \in \{1, \ldots, s\}$ such that $\deg(h_j f_j) = \deg(f)$. By consequence, $0 \neq \sum_{i=1}^s h_i f_i^h \in \mathcal{I}^h$ where $\mathcal{I}^h$ is the ideal generated by $\{f_1^h, \ldots, f_s^h\}$ and $\mathrm{LT}_{\mathrm{drl}}(f) = \mathrm{LT}_{\mathrm{drl}}\left(\sum_{i=1}^s h_i f_i^h\right)$. Thus, $\mathrm{in}_{drl}(\mathcal{I}) \subset \mathrm{in}_{drl}\left(\mathcal{I}^h\right)$. It is straightforward that $\mathrm{in}_{drl}\left(\mathcal{I}^h\right) \subset \mathrm{in}_{drl}(\mathcal{I})$ hence $\mathrm{in}_{drl}\left(\mathcal{I}^h\right) = \mathrm{in}_{drl}(\mathcal{I})$.

For all $g \in \mathrm{GL}(\mathbb{K}, n)$, since g is invertible the sequence $(g \cdot f_1, \ldots, g \cdot f_s)$ is also regular. Indeed, if there exists $i \in \{1, \ldots, s\}$ such that $g \cdot f_i$ is a divisor of zero in $\mathbb{K}[x_1, \ldots, x_n]/\langle g \cdot f_1, \ldots, g \cdot f_i \rangle$ then f_i is a divisor of zero in $\mathbb{K}[x_1, \ldots, x_n]/\langle f_1, \ldots, f_i \rangle$. Hence,

$$\mathrm{in}_{drl}(g \cdot \mathcal{I}) = \mathrm{in}_{drl}\left((g \cdot \mathcal{I})^h\right) .$$

Moreover, g is a linear change of variables thus it preserves the degree. Hence, for all $f \in \mathcal{I}$, we have $(g \cdot f)^h = g \cdot f^h$. Finally, let U_a be a Zariski open subset of $\mathrm{GL}(\mathbb{K}, n)$ such that for all $g \in U_a$, we have the equality $\mathrm{in}_{drl}\left(g \cdot \mathcal{I}^h\right) = \mathrm{Gin}(\mathcal{I}^h)$. Thus, for all $g \in U_a$, we then have $\mathrm{in}_{drl}(g \cdot \mathcal{I}) = \mathrm{in}_{drl}\left((g \cdot \mathcal{I})^h\right) = \mathrm{in}_{drl}\left(g \cdot \mathcal{I}^h\right) = \mathrm{Gin}(\mathcal{I}^h)$. □

Hence, from the previous proposition, for a random linear change of variables $g \in \mathrm{GL}(\mathbb{K}, n)$ we have $\mathrm{in}_{drl}(g \cdot \mathcal{I}) = \mathrm{Gin}\left(\mathcal{I}^h\right)$. Thus from Theorem 7.2, for all generators m of $\mathrm{in}_{drl}(g \cdot \mathcal{I})$ (*i.e.* m is a leading term of a polynomial in the DRL Gröbner basis of $g \cdot \mathcal{I}$) if x_n^t divides m and x_n^{t+1} does not divide m then for all $j < n$ we have $\frac{x_j}{x_n} m \in \mathrm{in}_{drl}(g \cdot \mathcal{I})$ if $t \not\equiv 0 \mod p$. Therefore, in the same way as for generic ideals, the multiplication matrix T_n of $g \cdot \mathcal{I}$ can be read from its DRL Gröbner basis. This is summarized in the following corollary.

Corollary 3. *Let $\mathbb{K}$ be an infinite field of characteristic $p \geq 0$. Let $\mathcal{I}$ be a radical ideal of $\mathbb{K}[x_1, \ldots, x_n]$. There exists a Zariski open subset U of $GL(\mathbb{K}, n)$ such that for all $g \in U$, the arithmetic complexity of computing the multiplication matrix by x_n of $g \cdot \mathcal{I}$ given its DRL Gröbner basis can be done without arithmetic operation. If $p > 0$ this is true only if $\deg_{x_n}(m) \not\equiv 0 \mod p$ for all $m \in E(g \cdot \mathcal{I})$. Consequently, under the same hypotheses, computing the LEX Gröbner basis of $g \cdot \mathcal{I}$ given its DRL Gröbner basis can be bounded by $O(\log_2(D)(D^\omega + n \log_2(D) D))$ arithmetic operations.*

Following this result, we propose another algorithm for polynomial systems solving.

7.2 Another algorithm for polynomial systems solving

Let $\mathcal{S} \subset \mathbb{K}[x_1, \ldots, x_n]$ be a polynomial system generating a radical ideal denoted $\mathcal{I}$. For any $g \in \mathrm{GL}(\mathbb{K}, n)$, from the solutions of $g \cdot \mathcal{I}$ one can easily recover the solutions of $\mathcal{I}$. Let U be the Zariski open subset of $\mathrm{GL}(\mathbb{K}, n)$ such that for all $g \in U$, $\mathrm{in}_{drl}(g \cdot \mathcal{I}) = \mathrm{Gin}(\mathcal{I}^h)$. If g is chosen in U then the multiplication matrix T_n can be computed very efficiently. Indeed, from Section 7.1 all monomials of the form $\epsilon_i x_n$ for $i = 1, \ldots, D$ are in B or in $E(g \cdot \mathcal{I})$ and their normal are easily known. Moreover, as mentioned in Section 2, there exists U' a the Zariski open subset of $\mathrm{GL}(\mathbb{K}, n)$ such that for all $g \in U'$ the ideal $g \cdot \mathcal{I}$ admits a LEX Gröbner basis in *Shape Position*. If g is also chosen in U' then we can use Algorithm 3 to compute the LEX Gröbner basis of $g \cdot \mathcal{I}$. Hence, we propose in Algorithm 5 a Las Vegas algorithm to solve the PoSSo problem. A Las Vegas algorithm is a randomized algorithm whose output (which can be *fail*) is always correct. The end of this section is devoted to evaluate its complexity and its probability of success *i.e.* when the algorithm does not return *fail*.

Algorithm 5 successes if the three following conditions are satisfied

Algorithm 5: Another algorithm for PoSSo.

Input : A polynomial system $\mathcal{S} \subset \mathbb{K}[x_1, \dots, x_n]$ generating a radical ideal.
Output: g in $\mathrm{GL}(\mathbb{K}, n)$ and the LEX Gröbner basis of $\langle g \cdot \mathcal{S} \rangle$ *i.e.* a univariate parametrization of the solutions of $\mathcal{S}$ or *fail*.

Choose randomly g in $\mathrm{GL}(\mathbb{K}, n)$;
Compute $\mathbb{G}_{\mathrm{drl}}$ the DRL Gröbner basis of $g \cdot \mathcal{S}$;
if T_n *can be read from* $\mathbb{G}_{drl}$ **then**
    Extract T_n from $\mathbb{G}_{\mathrm{drl}}$;
    From T_n and $\mathbb{G}_{\mathrm{drl}}$ compute $\mathbb{G}_{\mathrm{lex}}$ using Algorithm 3;
    if Algorithm 3 succeeds **then return** g and $\mathbb{G}_{\mathrm{lex}}$;
    else return *fail*;
else return *fail*;

1. $g \in \mathrm{GL}(\mathbb{K}, n)$ is chosen in a non empty Zariski open set U' such that for all $g \in U'$, $g \cdot \mathcal{I}$ has a LEX Gröbner basis in *Shape Position*;

2. $g \in \mathrm{GL}(\mathbb{K}, n)$ is chosen in a non empty Zariski open set U such that for all $g \in U$, $\mathrm{in}_{drl}(g \cdot \mathcal{I}) = \mathrm{Gin}(\mathcal{I}^h)$;

3. $p = 0$ or $p > 0$ and for all $m \in E(g \cdot \mathcal{I})$, $\deg_{x_n}(m) \not\equiv 0 \mod p$.

The existence of the non empty Zariski open subset U' is proven in [23]. Conditions (1) and (2) are satisfied if $g \in U \cap U'$. Since, U and U' are open and dense, $U \cap U'$ is also a non empty Zarisky open set.

7.2.1 Probability of success of Algorithm 5

Usually, the coefficient field of the polynomials is the field of rational numbers or a finite field. For fields of characteristic zero, if g is chosen randomly then the probability that the condition (1) and (2) be satisfied is 1. By consequence, the probability of success of Algorithm 3, in case of field of characteristic zero, is 1.

For finite fields $\mathbb{F}_q$, the Schwartz-Zippel lemma [39, 44] allows to bound the probability that the conditions (1) and (2) do not be satisfied by $\frac{d}{q}$ where d is the degree of the polynomial defining $U \cap U'$. Thus, in order to bound this failure probability we recall briefly how are constructed U and U'.

Construction of U'. Let $\mathcal{I} = \langle f_1, \dots, f_n \rangle$ be a radical ideal of $\mathbb{K}[x_1, \dots, x_n]$. Since $\mathcal{I}$ is radical, all its solutions are distinct. Therefore, let $\{a_i = (a_{i,1}, \dots, a_{i,n}) \in \overline{\mathbb{K}}^n \mid f_j(a_1, \dots, a_n) = 0,\ j = 1, \dots, n\}$ be the set of solutions of $\mathcal{I}$ (recall that its cardinality is D). Let g be a given matrix in $\mathrm{GL}(\mathbb{K}, n)$. We denote by $v_i = (v_{i,1}, \dots, v_{i,n})$ the point obtained after transformation of a_i by g, *i.e* $v_i = g \cdot a_i^t$. To ensure that $g \cdot \mathcal{I}$ admits a LEX Gröbner basis in *Shape Position*, g should be such that $v_{i,n} \neq v_{j,n}$ for all couples of integers (i, j) verifying $1 \leq j < i \leq D$. Hence, let $\mathbf{g} = (\mathbf{g}_{i,j})$ be a $(n \times n)$ matrix of unknowns, the polynomial $P_{U'}$ defining the Zariski open subset U' is then given as the determinant of the Vandermonde matrix associated to $\mathbf{v}_{i,n}$ for $i = 1, \dots, D$ where $\mathbf{v}_i = (\mathbf{v}_{i,1}, \dots, \mathbf{v}_{i,n}) = \mathbf{g} \cdot a_i^t$. Therefore, we know exactly the degree of $P_{U'}$ which is $\frac{D(D-1)}{2}$.

Construction of U. The Zariski open subset U is constructed as the intersection of Zariski open subsets $U_1, \dots, U_\delta$ of $\mathrm{GL}(\mathbb{K}, n)$ where δ is the maximum degree of the generators of $\mathrm{Gin}(\mathcal{I}^h)$. Let d be a fixed degree. Let $\mathbb{K}[x_1, \dots, x_n]_d = R_d$ be the set of homogeneous polynomials of degree d of $\mathbb{K}[x_1, \dots, x_n]$. Let $\mathbb{G}_{\mathrm{drl}}^h$ be the DRL Gröbner basis of $\mathcal{I}^h$ and let $\{f_1, \dots, f_{t_d}\} = \mathbb{G}_{\mathrm{drl}}^h \cap R_d$ be a vector basis of $\mathcal{I}_d^h = \mathcal{I}^h \cap R_d$.

Let $\mathbf{g} = (\mathbf{g}_{i,j})$ be a $(n \times n)$ matrix of unknowns and let M be a matrix representation of the map $\mathcal{I}_d^h \to \mathbf{g} \cdot \mathcal{I}_d^h$ defined as follow:

$$M = (M_{i,j}) = \begin{array}{c} \begin{array}{ccc} m_1 & \cdots & m_N \end{array} \\ \boxed{\begin{array}{ccc} \star & \cdots & \star \\ \vdots & \ddots & \vdots \\ \star & \cdots & \star \end{array}} \end{array} \begin{array}{c} \\ \mathbf{g} \cdot f_1 \\ \vdots \\ \mathbf{g} \cdot f_{t_d} \end{array}$$

where $M_{i,j}$ is the coefficient of m_j in $\mathbf{g} \cdot f_i$ and $\{m_1, \ldots, m_N\}$ is the set of monomials in R_d. In [5, 15], the polynomial P_{U_d} defining U_d is constructed as a particular minor of size t_d of M. Since each coefficient in M is a polynomial in $\mathbb{K}[\mathbf{g}_{1,1}, \ldots, \mathbf{g}_{n,n}]$ of degree d, the degree of P_{U_d} is $d \cdot t_d$. Finally, since U_d is open and dense for all $d = 1, \ldots, \delta$ we deduce that $U = \cap_{i=1}^{\delta} U_d$ is a non empty Zariski open set whose defining polynomial, P_U, is of degree $\sum_{d=1}^{\delta} d \cdot t_d \leq \delta \sum_{i=1}^{\delta} t_d$. Moreover, the number of polynomials in $\mathbb{G}_{\text{drl}}^h$ is bounded by nD. Thus, $\deg(P_U) \leq \delta n D$.

For ideals generated by a regular sequence $(f_1, \ldots, f_n)$, thanks to the Macaulay's bound, δ can be bounded by $\sum_{i=1}^{n} (\deg(f_i) - 1) + 1$. Note that the Macaulay's bound gives also a bound on $\deg_{x_n}(m)$ for all $m \in E(g \cdot \mathcal{I})$. To conclude, the probability that conditions (1) and (2) be satisfied is greater than $1 - \frac{1}{q} \left(\frac{D(D-1)}{2} + \left(\sum_{i=1}^{n} (\deg(f_i) - 1) + 1 \right) nD \right)$; and if $p > \sum_{i=1}^{n} (\deg(f_i) - 1) + 1$ then condition (3) is satisfied.

7.2.2 Complexity of Algorithm 5

As previously mentioned, the matrix T_n can be read from $\mathbb{G}_{\text{drl}}$ (test in Line 3 of Algorithm 5) if all the monomials of the form $\epsilon_i x_n$ are either in B or in $E(\langle \mathbb{G}_{\text{drl}} \rangle)$. Let $F_n = \{\epsilon_i x_n \mid i = 1, \ldots, D\}$, the test in Line 3 is equivalent to test if $F_n \subset B \cup E(\langle \mathbb{G}_{\text{drl}} \rangle)$. Since F_n contains exactly D monomials and $B \cup E(\langle \mathbb{G}_{\text{drl}} \rangle)$ contains at most $(n+1)D$ monomials; in a similar way as in Lemma 2 testing if $F_n \subset B \cup E(\langle \mathbb{G}_{\text{drl}} \rangle)$ can be done in at most $O(nD^2)$ elementary operations which can be decreased to $O(D)$ elementary operations if we use a hash table. Hence, the cost of computing B, F_n (see Lemma 2) and the test in Line 3 of Algorithm 5 are negligible in comparison to the complexity of Algorithm 3. Hence, the complexity of Algorithm 5 is given by the complexity of F_5 algorithm to compute the DRL Gröbner basis of $g \cdot \mathcal{I}$ and the complexity of Algorithm 3 to compute the LEX Gröbner basis of $g \cdot \mathcal{I}$. From [31], the complexities of computing the DRL Gröbner basis of $g \cdot \mathcal{I}$ or $\mathcal{I}$ are the same. Since it is straightforward to see that the number of solutions of these two ideals are also the same we obtain the second main result of the paper.

Theorem 7.3. *Let $\mathbb{K}$ be the rational field $\mathbb{Q}$ or a finite field $\mathbb{F}_q$ of sufficiently large characteric p. Let $\mathcal{S} = \{f_1, \ldots, f_n\} \subset \mathbb{K}[x_1, \ldots, x_n]$ be a polynomial system generating a radical ideal $\mathcal{I} = \langle \mathcal{S} \rangle$ of degree D. If the sequence $(f_1, \ldots, f_n)$ is a regular sequence such that the degree of each polynomial is uniformly bounded by a fixed or non fixed parameter d then there exists a Las Vegas algorithm which computes the univariate polynomial representation of the roots of $\mathcal{S}$ in $O(d^{\omega n} + \log_2(D)(D^\omega + n \log_2(D) D))$ arithmetic operations.*

As previously mentioned, the Bézout's bound allows to bound the number of solutions D by the product of the degrees of the input equations. Since this bound is generically reached we get the following corollary.

Corollary 4. *Let $\mathbb{K}$ be the rational field $\mathbb{Q}$ or a finite field $\mathbb{F}_q$ of sufficiently large characteric p. Let $\mathcal{S} = \{f_1, \ldots, f_n\} \subset \mathbb{K}[x_1, \ldots, x_n]$ be a generic polynomial system generating an ideal $\mathcal{I} = \langle \mathcal{S} \rangle$ of degree D. If the degree of each polynomial in $\mathcal{S}$ is uniformly bounded by a fixed or non fixed parameter d then there exists a Las Vegas algorithm which computes the univariate polynomial representation of the roots of $\mathcal{S}$ in*

$\widetilde{O}(D^\omega)$ arithmetic operations where the notation $\widetilde{O}$ means that we neglect logarithmic factors in D and polynomial factors in n.

8 Acknowledgments

The authors would like to thanks André Galligo and Daniel Lazard for fruitful discussions about generic initial ideals and Vanessa Vitse for providing us with an example for which the randomization is required to *"freely"* construct the multiplication matrix T_n.

References

[1] M. Bardet, J.-C. Faugère, B. Salvy, and B. Yang. Asymptotic behaviour of the degree of regularity of semi-regular polynomial systems. In P. Gianni, editor, *The Effective Methods in Algebraic Geometry Conference, Mega 2005*, pages 1 –14, May 2005.

[2] S. Basu, M.-F. Roy, M. Safey El Din, and É. Schost. A baby step-giant step roadmap algorithm for general algebraic sets. *CoRR*, abs/1201.6439, 2012.

[3] W. Baur and V. Strassen. The complexity of partial derivatives. *Theoretical computer science*, 22(3):317–330, 1983.

[4] D. Bayer and M. Stillman. A criterion for detecting m-regularity. *Inventiones mathematicae*, 87(1):1–11, 1987.

[5] D. Bayer and M. Stillman. A theorem on refining division orders by the reverse lexicographic order. *Duke Mathematical Journal*, 55(2):321–328, 1987.

[6] 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*, ISSAC '94, pages 129–133, New York, NY, USA, 1994. ACM.

[7] A. Bostan, B. Salvy, and E. Schost. Fast algorithms for zero-dimensional polynomial systems using duality. *Applicable Algebra in Engineering, Communication and Computing*, 14(4):239–272, 2003.

[8] R. P. Brent, F. G. Gustavson, and D. Y. Yun. Fast solution of Toeplitz systems of equations and computation of Padé approximants. *Journal of Algorithms*, 1(3):259–295, 1980.

[9] J. Buchmann, A. Pyshkin, and R.-P. Weinmann. A zero-dimensional Gröbner basis for AES-128. In *Fast Software Encryption*, pages 78–88. Springer, 2006.

[10] J. Bunch and J. Hopcroft. Triangular factorization and inversion by fast matrix multiplication. *Mathematics of Computation*, 28(125):231–236, 1974.

[11] J. F. Canny. Computing roadmaps of general semi-algebraic sets. *Comput. J.*, 36(5):504–514, 1993.

[12] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms: an Introduction to Computational Algebraic Geometry and Commutative Algebra*, volume 10. Springer, 2007.

[13] R. S. Datta. Universality of Nash equilibria. *Mathematics of Operations Research*, 28(3):424–432, 2003.

[14] M. De Boer and R. Pellikaan. *Some Tapas of Computer Algebra*, volume 4, chapter Gröbner Bases for Codes. Springer, 1999.

[15] D. Eisenbud. *Commutative Algebra with a View Toward Algebraic Geometry*. Springer, 1995.

[16] 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.

[17] J.-C. Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). In *Proceedings of the 2002 international symposium on Symbolic and algebraic computation*, ISSAC '02, pages 75–83, New York, NY, USA, 2002. ACM.

[18] J.-C. Faugère, P. Gaudry, L. Huot, and G. Renault. Fast change of ordering with exponent ω. *ACM Commun. Comput. Algebra*, 46:92–93, September 2012.

[19] 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(4):329–344, 1993.

[20] J.-C. Faugère and C. Mou. Sparse FGLM algorithms. `http://hal.inria.fr/hal-00807540`.

[21] J.-C. Faugère and C. Mou. Fast algorithm for change of ordering of zero-dimensional Gröbner bases with sparse multiplication matrices. In *Proceedings of the 36th international symposium on Symbolic and algebraic computation*, ISSAC '11, pages 115–122, New York, NY, USA, 2011. ACM.

[22] A. Galligo. *A Propos du Théorème de Préparation de Weierstrass*. PhD thesis, Institut de Mathématique et Sciences Physiques de l'Université de Nice, 1973.

[23] P. Gianni and T. Mora. Algebraic solution of systems of polynomial equations using Gröbner bases. In *Applied Algebra, Algebraic Algorithms and Error Correcting Codes, Proceedings of AAECC-5, volume 356 of LNCS*, pages 247–257. Springer, 1989.

[24] A. Greuet and M. Safey El Din. Deciding reachability of the infimum of a multivariate polynomial. In *ISSAC 2011—Proceedings of the 36th International Symposium on Symbolic and Algebraic Computation*, pages 131–138. ACM, New York, 2011.

[25] E. Jonckheere and C. Ma. A simple Hankel interpretation of the Berlekamp-Massey algorithm. *Linear Algebra and its Applications*, 125:65–76, 1989.

[26] A. Joux. A new index calculus algorithm with complexity $L(1/4 + o(1))$ in very small characteristic. Cryptology ePrint Archive, Report 2013/095, 2013. `http://eprint.iacr.org/`.

[27] W. Keller-Gehrig. Fast algorithms for the characteristic polynomial. *Theor. Comput. Sci.*, 36:309–317, June 1985.

[28] H. Kobayashi, T. Fujise, and A. Furukawas. Solving systems of algebraic equations by a general elimination method. *Journal of Symbolic Computation*, 5(3):303 – 320, 1988.

[29] Y. N. Lakshman. On the complexity of computing a Gröbner basis for the radical of a zero dimensional ideal. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, STOC '90, pages 555–563, New York, NY, USA, 1990. ACM.

[30] Y. N. Lakshman and D. Lazard. On the complexity of zero-dimensional algebraic systems. In *Effective methods in algebraic geometry*, volume 94, page 217. Birkhauser, 1991.

[31] D. Lazard. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In J. van Hulzen, editor, *Computer Algebra*, volume 162 of *Lecture Notes in Computer Science*, pages 146–156. Springer Berlin / Heidelberg, 1983.

[32] P. Loustaunau and E. V. York. On the decoding of cyclic codes using Gröbner bases. *Applicable Algebra in Engineering, Communication and Computing*, 8(6):469–483, 1997.

[33] F. S. Macaulay. *The algebraic theory of modular systems*. Cambridge Mathematical Library. Cambridge University Press, Cambridge, 1994. Revised reprint of the 1916 original, With an introduction by Paul Roberts.

[34] J. Massey. Shift-register synthesis and BCH decoding. *IEEE Transactions on Information Theory*, 15(1):122–127, 1969.

[35] G. Moreno-Socias. Degrevlex Gröbner bases of generic complete intersections. *Journal of Pure and Applied Algebra*, 180(3):263–283, 2003.

[36] B. Mourrain and V. Y. Pan. Asymptotic acceleration of solving multivariate polynomial systems of equations. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 488–496. ACM, 1998.

[37] V. Y. Pan. Univariate polynomials: Nearly optimal algorithms for numerical factorization and root-finding. *Journal of Symbolic Computation*, 33(5):701 – 733, 2002.

[38] K. Pardue. *Nonstandard Borel-Fixed Ideals*. PhD thesis, Brandeis University, 1994.

[39] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, Oct. 1980.

[40] B. Sturmfels. *Solving Systems of Polynomial Equations*, volume 97. American Mathematical Society, 2002.

[41] V. Vassilevska Williams. Multiplying matrices faster than Coppersmith-Winograd. In *Proceedings of the 44th symposium on Theory of Computing*, pages 887–898. ACM, 2012.

[42] J. Von Zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 2003.

[43] D. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Transactions on Information Theory*, 32(1):54–62, 1986.

[44] R. Zippel. Probabilistic algorithms for sparse polynomials. In E. Ng, editor, *Symbolic and Algebraic Computation*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer Berlin Heidelberg, 1979.