7th International Conference on
Combinatorics, Cryptography, Computer Science and Computing
November. 16-17, 2022

Generalization of the F_4 Algorithm to Parametric Polynomial Ideals

Mahdi Dehghani Darmian[1]
Department of Mathematics, Technical and Vocational University (TVU),
Tehran, Iran

Amir Hashemi
Department of Mathematical Sciences, Isfahan University of Technology,
Isfahan, 84156-83111, Iran

Abstract

In this paper, we design a parametric version of the F_4 algorithm (so-called PF_4) which can be considered as a expantion of Faugère's F_4 algorithm [5] to the parametric polynomial ideals. Our idea is based on the F_4 algorithm and the Montes DisPGB algorithm [10]. Also, we apply the parametric linear algebra methods developed in [3]. The input of the PF_4 algorithm is a parametric polynomial ideal and two monomial orderings on the variables and the parameters. It returns a Gröbner system of the ideal with respect to a compatible elimination product of the given monomial orderings. We have implemented our new algorithm in Maple and give timings and used momory to compare its performance with our implementation of the DisPGB algorithm [10] and the Kapur et al. algorithm namely PGBMain [8].

Keywords: Gröbner systems, F_4 algorithm, PF_4 algorithm, PGBMain algorithm, DisPGB algorithm.

AMS Mathematical Subject Classification [2010]: 13P10, 68W30.

---

1 Introduction

Gröbner bases as a particular kind of generating set of the polynomial ideals are a powerful computational tool in computer algebra with many important applications in Mathematics, science, and engineering. These bases together with the first algorithm to compute them were introduced by Buchberger in his Ph.D. thesis under the supervision of Gröbner [2]. Later on, he accelerated his algorithm by propounding two criteria to remove redundant reductions [1]. Nevertheless, Buchberger's algorithm is so time consuming and that is why Lazard in 1983 proposed an algorithm for computing Gröbner bases, by using linear algebra methods [9]. In 1999 and 2002 Faugère presented, respectively, his two well-known F4 [5] and F5 [6] algorithms for computing Gröbner bases. The F4 algorithm uses the same mathematical concepts as the Buchberger algorithm, but computes many normal forms simultaneously by forming a generally sparse matrix and using fast linear algebra to perform the reductions.

In this paper, we adapt the F_4 algorithm to compute *Gröbner systems* (as an extension of Gröbner bases) for parametric polynomial ideals. A Gröbner system is a finite set of triples (so-called branches); each branch contains a parametric constraints (a couple of null and non-null parametric sets) and also a set of polynomials so that for any

[1] speaker

specialization there is a branch so that the specialization satisfies its constraints, and the specialized polynomial set forms a Gröbner basis for the parametric ideal under the substitution of the values of the parameters. The concept of *Gröbner system* was introduced by Weispfenning in [12]. He proved also the existence of a Gröbner system for any given parametric polynomial ideal [12, Proposition 3.4 and Theorem 2.7] and presented the first algorithm to compute it [12, Theorem 3.6]. In 2002, Montes [10] propounded a more efficient algorithm (DisPGB) for computing Gröbner systems (see also [4, 7]). In [11] Suzuki and Sato proposed an effective improvement for computing Gröbner systems. The Suzuki-Sato algorithm utilized recursively computations of reduced Gröbner bases in an extension of the base polynomial ring. Finally, Kapur et al. in 2010 designed an efficient algorithm (PGBMain algorithm [8]) for computing Gröbner systems by using combination of the Weispfenning [13] and the Suzuki-Sato algorithms. The PGBMain algorithm at each iteration computes the Gröbner basis over a polynomial ring in terms of the variables and the parameters. Therefore, this step may be very expensive and time consuming in practice, because the complexity of Gröbner basis computation is extremely impressed by the number of variables and degree of the given polynomials. Therefore, it is important to design an efficient algorithm to reduces the computation in a polynomial ring in terms of only the variables. On the other hand, the DisPGB algorithm works in a polynomial ring in terms of only the variables. At this algorithm, like the Buchberger's algorithm, a parametric S-polynomials is computed and if its remainder is non zero then it is added to the basis set (see also [7]). In consequence, DisPGB creates new branches when a new polynomial with an undecidable coefficient is constructed and this may leads to many number of branches which can cause the inefficiency of the algorithm in practice. In order to prevent this problem, we present a parametric version of the F_4 algorithm. Based on parametric linear algebra technique and using the basic ideas from the DisPGB structure [10], we propose a parametric F_4 algorithm to compute Gröbner systems for parametric polynomial ideals. Our new presented algorithm along with the the PGBMain and DiSPGB algorithms have been implemented in Maple and their efficiency is discussed on a diverse set of parametric polynomial ideals.

In the following, we give a very brief review of the basic notations and definitions relating to Gröbner bases and Gröbner systems.

Throughout this paper, we consider $\mathcal{R} = \mathbb{K}[x_1, \ldots, x_n]$ the polynomial ring in terms of $x_1, \ldots, x_n$ over a field $\mathbb{K}$. Let $\mathcal{I} = \langle f_1, \ldots, f_k \rangle \subset \mathcal{R}$ be the polynomial ideal generated by the f_i's. We consider a monomial ordering $\prec$ on the set of all monomials (power products of the x_i's) of $\mathcal{R}$. For any $f \in R$, the *leading monomial* of f, denoted by $\mathrm{LM}_\prec(f)$, is the greatest monomial (with respect to $\prec$) appearing in f and its coefficient is the *leading coefficient* of f which denoted by $\mathrm{LC}_\prec(f)$. The *leading term* of f with respect to $\prec$ is the product $\mathrm{LT}_\prec(f) = \mathrm{LC}_\prec(f)\mathrm{LM}_\prec(f)$. The *leading monomial ideal* of $\mathcal{I}$ is defined to be $\mathrm{LM}_\prec(\mathcal{I}) = \langle \mathrm{LM}_\prec(f) \mid f \in \mathcal{I} \rangle$. A finite subset $\{g_1, \ldots, g_m\} \subset \mathcal{I}$ is called a *Gröbner basis* for $\mathcal{I}$ with respect to $\prec$ if $\mathrm{LM}_\prec(\mathcal{I}) = \langle \mathrm{LM}_\prec(\mathrm{g}_1), \ldots, \mathrm{LM}_\prec(\mathrm{g}_\mathrm{m}) \rangle$.

Now consider $\mathcal{S} = \mathbb{K}[\mathbf{a}, \mathbf{x}]$ where $\mathbb{K}$ is an arbitrary field, $\mathbf{a} = a_1, \ldots, a_m$ is a sequence of parameters and $\mathbf{x} = x_1, \ldots, x_n$ is a sequence of variables. Let $\prec_\mathbf{x}$ be a monomial order on the variables and $\prec_\mathbf{a}$ be a monomial order on the parameters. The product of $\prec_\mathbf{x}$ and $\prec_\mathbf{a}$ gives rise to an ordering on $\mathcal{S}$, denoted by $\prec_{\mathbf{x},\mathbf{a}}$ which is defined as follows: For all $\alpha, \beta \in \mathbb{N}^n$ and $\gamma, \delta \in \mathbb{N}^m$, $\mathbf{x}^\alpha \mathbf{a}^\gamma \prec_{\mathbf{x},\mathbf{a}} \mathbf{x}^\beta \mathbf{a}^\delta \iff \mathbf{x}^\alpha \prec_\mathbf{x} \mathbf{x}^\beta$or ($\mathbf{x}^\alpha = \mathbf{x}^\beta$and $\mathbf{a}^\gamma \prec_\mathbf{a} \mathbf{a}^\delta$). Let us consider $\sigma : \mathbb{K}[\mathbf{a}] \to \overline{\mathbb{K}}$ as a specialization of parameters where $\overline{\mathbb{K}}$ is the algebraic closure of $\mathbb{K}$. This morphism can be considered as a substitution of existent parameters in $f \in \mathbb{K}[\mathbf{a}]$ with an elements of $\overline{\mathbb{K}}^m$. Also, for a finite set $F \subset \mathcal{R}$, we call $\mathbb{V}(F)$ the variety of F which is the set of common zeros of F. Now, we are ready to recall the definition of a Gröbner system for a parametric polynomial ideal.

Definition 1.1. Let $F \subset \mathcal{S}$ and $\mathcal{G} = \{(G_i, N_i, W_i)\}_{i=1}^{\ell}$ be a finite triples set where $N_i, W_i \subset \mathbb{K}[\mathbf{a}]$ and $G_i \subset \mathcal{S}$ are finite for $i = 1, \ldots, \ell$. The set $\mathcal{G}$ is called a *Gröbner system* of $\langle F \rangle$ w.r.t. $\prec_{\mathbf{x},\mathbf{a}}$ on $V \subseteq \overline{\mathbb{K}}^m$ if for any i we have

- For any specialization $\sigma : \mathbb{K}[\mathbf{a}] \to \overline{\mathbb{K}}$ satisfying (N_i, W_i) the set $\sigma(G_i) \subset \overline{\mathbb{K}}[\mathbf{x}]$ is a Gröbner basis of $\langle \sigma(F) \rangle$ w.r.t. $\prec_{\mathbf{x}}$. (We say that σ satisfies (N_i, W_i) if $\sigma(p) = 0$ for all $p \in N_i$ and $\sigma(q) \neq 0$ for some $q \in W_i$)

- $V \subseteq \bigcup_{i=1}^{\ell} \mathbb{V}(N_i) \setminus \mathbb{V}(W_i)$

Each (N_i, W_i, G_i) is called a branch of the Gröbner system $\mathcal{G}$ and we can consider (N_i, W_i) as a condition set which N_i is the null condition set and W_i the non-null condition set. Furthermore, $\mathcal{G}$ is a Gröbner system of F if $V = \overline{\mathbb{K}}^m$.

Example 1.2. Let $F = \{(1-c)y - ax^2, x + by^2\} \subset \mathbb{K}[a, b, c, x, y]$ where a, b, c are parameters and x, y are variables. Using our implementation of PGBMain algorithm in Maple, we obtain the following CGS for $\langle F \rangle$ w.r.t. the product ordering $y \prec_{lex} x$ and $c \prec_{lex} b \prec_{lex} a$

$$\begin{cases} ([\,], & [ab^2], & [ab^2y^4 - y + cy, x + by^2]) \\ ([ab^2], & [c-1], & [cy - y, x + by^2]) \\ ([c-1, ab^2], & [\,], & [x + by^2]). \end{cases}$$

For instance, if $a = 2, b = 0$ and $c = 3$ then the second branch corresponds to these values of parameters. Therefore, $\{x, y\}$ will be a Gröbner basis for the ideal $\langle F \rangle \mid_{a=2,b=0,c=3} = \langle -2y - 2x^2, x \rangle$.

2 Main results

The strength point of the Faugère's F4 algorithm [5] compared to the Buchberger algorithm is the use of row-reduction techniques on the *sparse matrix* to perform the reductions of several S-polynomial, simultaneously. In this section, we present a parametric F_4 algorithm which can be considered as a generalization of the F_4 algorithm to polynomial ideals with parametric coefficients. At each iteration of the F_4 algorithm we shall perform linear and non-linear reductions, and it is non-trivial to handle the parametric variants of all these reductions. To resolve this issue ,we apply the GES algorithm [3] with slight modifications. This algorithm computes a Gaussian elimination system for a parametric matrix (equivalently a parametric linear system corresponding to the input matrix). However, we utilize this algorithm on non-linear polynomials to make a linear inter-reduction, and we look for their Gaussian forms according to parametric constraints. In this direction, we shall linearize the input polynomials by replacing each monomial appearing in the polynomials by new variables. The main engine of the GES algorithm is the LDS algorithm [3] which discusses the dependency of a linear parametric polynomial with respect to a given set of parametric polynomials without the use of Gröbner systems. More precisely, the input of LDS algorithm is (N, W, F, f) where (N, W) is a pair of condition sets, F is a set of linear parametric polynomials (which forms a parametric Gröbner basis with respect to the given parametric constraint) and f is a linear parametric polynomial. LDS algorithm returns a finite set of triples of the form $(N_i, W_i, [flag, Q, g])$ where (N_i, W_i) is a pair of condition sets, flag is a Boolean variable, Q represents the quotients of the division and g is the normal form of f with respect to F. If flag is true then $g = 0$ and in consequence f is linear dependent on F with respect to (N_i, W_i), and if it is false then f is linear independent modulo F with respect to (N_i, W_i).

Algorithm 1 LDS (Linear Dependency System)

Require: $G \subset \mathcal{S}$; a linear set which is a reduced Gröbner basis w.r.t. the product of the monomial orderings $\prec_{\mathbf{x}}$ and $\prec_{\mathbf{a}}$ provided that a conditions pair (N, W) is satisfied and $g \in \mathcal{S}$; a parametric linear polynomial
Ensure: A linear dependency system of g on (N, W, G)

```
Sys:= {}
f, Q :=NormalForm(g,GröbnerBasis(N, ≺_a), ≺_a)
f', Q' :=NormalForm(f, G, ≺_x)
if f' = 0 then
    Sys:=Sys ⋃{(N, W, [true, Q', 0])}
else
    A := {a_{i_1}, ..., a_{i_t}} where f' = a_{i_1} x_{i_1} + ··· + a_{i_t} x_{i_t} with a_{i_j} ≠ 0 and x_{i_1} >_x ··· >_x x_{i_t}
    for j from 1 to t do
        if a_{i_j} is not constant then
            Sys:=Sys ⋃{(N ∪ {a_{i_1}, ..., a_{i_{j-1}}}, W ∪ {a_{i_j}}, [false, Q', f'|_{a_{i_1}=0,...,a_{i_{j-1}}=0}])}
        else
            Sys:=Sys ⋃{(N ∪ {a_{i_1}, ..., a_{i_{j-1}}}, W, [false, Q', f'|_{a_{i_1}=0,...,a_{i_{j-1}}=0}])}
            Return(Sys)
        end if
    end for
    Sys:=Sys ⋃{(N ⋃ A, W, [true, Q', 0])}
end if
Return(Sys)
```

Theorem 2.1. LDS *algorithm terminates and is correct.*

Proof. The termination of these algorithm is trivial. The correctness of LDS algorithm comes from the fact that g is dependent on G iff either $f' = 0$ or all the coefficients of f' are null. If $f' \neq 0$, we add the coefficients of f' (which are polynomials in $\mathbb{K}[\mathbf{a}]$) into N and verify the consistency of the new conditions pairs. □

Using LDS algorithm, we are aspiring to propose an efficient algorithm to compute a Gaussian elimination system for a set of parametric polynomials. Below, the variable Sys is initialized to the empty set, and finally, it is the output Gaussian elimination system. In more detail, each saved segment in Sys is a triple (N, W, G) where (N, W) is a pair of condition sets and G is a Gaussian elimination form of the input parametric polynomials set with respect to (N, W).

Algorithm 2 GES (Gaussian Elimination System)

Require: $N \subset \mathbb{K}[\mathbf{a}]$; null condition set, $W \subset \mathbb{K}[\mathbf{a}]$; non-null condition set, $F \subset \mathbb{K}[\mathbf{a}, \mathbf{x}]$; a parametric polynomial set
Ensure: A Gaussian elimination system of F according to N and W

```
Sys:= {}
M := Mon(F) = [m_1, ..., m_t] (the set of all monomials in terms of the x_i's appearing in F)
[Y_1, ..., Y_t] :=A list of tag variables corresponding to Mon(F)
L := φ(F) where φ is a linear map sending each m_i into Y_i
A := {(N, W, {}, L[1], L)}
while A ≠ {} do
    a := A[1] and A := A \ {a}
    if a[5] = {} then
        G := φ^{-1}(a[3])
        Sys:=Sys ⋃{(a[1], a[2], G)}
    else
        G := a[5] \ {a[4]}
        g := G[1]
        P :=LDS(a[1], a[2], a[3], a[4])
        for i from 1 to |P| do
            Let P[i] = (N_1, W_1, [flag, Q, f])
            if flag = true then
                A := A ∪ {(N_1, W_1, a[3], g, G)}
            else
                A := A ∪ {(N_1, W_1, a[3] ∪ {f}, g, G)}
            end if
        end for
    end if
end while
Return (Sys)
```

Theorem 2.2. *The GES algorithm terminates in finitely many steps and is correct.*

Proof. The termination of the GES algorithm is a direct consequence of the termination of the LDS algorithm and the finiteness of F. Also, the correctness of the LDS algorithm warrantes the correctness of this algorithm. More precisely, by the structure of the algorithm, we discuss a new polynomial $f \in F$ using the LDS algorithm. If it is linear dependent on the computed basis, then it is removed. Otherwise, its normal form with respect to the computed basis is added into the basis. Thus, each branch contains a Gaussian elimination form of the input parametric polynomial set according to the corresponding parametric constraint. □

Example 2.3. Let us consider $F = \{ax^2 + by + 1, cz^3 + (a-1)y - b, (a-b)y^2 + (c-1)xy - 2\} \subset \mathbb{K}[a,b,c][x,y,z]$ as a set of parametric polynomials. Using the GES algorithm, we get the following Gaussian elimination system for F when $(N, W) = ([a-1],[c])$.

$$\begin{cases} ([a-1], & [c, c-1], & [x^2+by+1, cz^3-b, (1-b)y^2+cxy-xy-2]), \\ ([c-1, a-1], & [b-1], & [x^2+by+1, z^3-b, (1-b)y^2-2]), \\ ([c-1, b-1, a-1], & [\ ], & [x^2+y+1, z^3-1, -2]). \end{cases}$$

We are willing now to present a parametric variant of the F_4 algorithm, so-called PF_4 for the computation of Gröbner systems for parametric polynomial ideals. This algorithm receives a parametric polynomial set F and two monomial orderings on variables and parameters and returns a Gröbner system for the ideal generated by F. Below, the notion OUTSYS stands for a global variable which is initialized to empty sequence and at each iteration of two algorithms PF_4 and PF_4Basis, some new branches are added to this sequence, and finally it is a Gröbner system of the input ideal.

Algorithm 3 PF_4

```
Require: F ⊂ K[a, x] = K[a_1, ..., a_m, x_1, ..., x_n], <_x, <_a; two monomial orderings
Ensure: G; A Gröbner system of ⟨F⟩ with respect to <_{x,a}
  OUTSYS := NULL
  A :=GES([ ], [ ], F)
  for (N_n, W_n, F_n) ∈ A do
    if F_n = [ ] then
      OUTSYS:= OUTSYS, (N_n, W_n, [ ])
    end if
    if there is any constant or non-zero parameter in F_n then
      OUTSYS:= OUTSYS, (N_n, W_n, [1])
    else
      t := |F_n| (the cardinality of F_n)
      B := [[{i, j}, deg(lcm(LM_{<x}(F_n[i]), LM_{<x}(F_n[j])))] | 1 ≤ i < j ≤ t, gcd(LM_{<x}(F_n[i]), LM_{<x}(F_n[j])) ≠ 1]
      if B = [ ] then
        OUTSYS:= OUTSYS, (N_n, W_n, F_n)
      else
        SYS := [[N_n, W_n, F_n, t, B]]
      end if
    end if
    PF4Basis(SYS)
  end for
```

In the following we describe the PF_4Basis algorithm which is the engine of the PF_4 algorithm. In this algorithm, the NewPolys is a procedure which receives two lists of polynomials F, G and returns the list of those polynomials $f \in F$ such that $\mathrm{LM(f)} \notin \langle \mathrm{LM(G)} \rangle$. Also, NormalSet$(F, G, \prec) = \{\overline{f}^G_\prec | f \in F\}$ where $\overline{f}^G_\prec$ is a remainder of f on division by G with respect to $\prec$. Moreover, in order to enhance the efficiency of the PF_4Basis algorithm, we keep track of the computations by saving any branch $sys \in SYS$ in the PF_4Basis algorithm of the form $sys = (a[1], \ldots, a[5])$ containing the following information:

- $a[1]$: The null condition set
- $a[2]$: The non-null condition set
- $a[3]$: The set of polynomials which forms a Gröbner basis for $\langle F \rangle$ with respect to $\prec_{\mathbf{x}}$
- $a[4]$: The cardinality of $a[3]$
- $a[5]$: A list of pairs so that the first component each element is a pair $\{i, j\}$ and second component is $\deg(\mathrm{lcm}(\mathrm{LM}_{\prec_x}(a[3][i]), \mathrm{LM}_{\prec_x}(a[3][j])))$.

Algorithm 4 PF_4Basis

Require: $N \subset \mathbb{K}[\mathbf{a}]$; null condition set, $W \subset \mathbb{K}[\mathbf{a}]$; non-null condition set, $F \subset \mathbb{K}[\mathbf{a},\mathbf{x}]$; t; the cardinality of F and $Cpairs$; a list of pairs so that the first component is a pair of integers $\{i, j\}$ and second component is $\deg(\mathrm{lcm}_{\prec_{\mathbf{x}}}(\mathrm{LM}(F[i]), \mathrm{LM}_{\prec_{\mathbf{x}}}(F[j])))$
Ensure: Decomposing the space of parameters into a finite set of parametric cells and for each cell associating a finite set of parametric polynomials

```
B := Cpairs
while SYS ≠ [ ] do
  sys := SYS[1] and remove it from SYS
  G := sys[3]
  B := sys[5]
  Select Bp ⊆ B using degree-normal selection strategy
  B := B \ Bp
  Bsys := B
  L := { lcm(LM(f_i),LM(f_j))/LM(f_i) . f_i,  lcm(LM(f_i),LM(f_j))/LM(f_j) . f_j | {i, j} ∈ Bp}
  H := ComputeM(L, G)
  Ges := GES(sys[1], sys[2], H)
  allNPi := [NewPolys(Ges[i][3], H), i = 1, ..., |Ges|]
  for j from 1 to |allNPi| do
    t := sys[4]
    G := NormalSet(sys[3], Ges[j][1], <_a);
    B := Bsys
    for ℓ from 1 to |allNPi[j]| do
      t := t + 1
      B := Update(allNPi[j][ℓ], G, B)
      G := [op(G), allNPi[j][ℓ]]
      if B = [ ] then
        OUTSYS:= OUTSYS, (Ges[j][1], Ges[j][2], G)
      else
        SYS := [op(SYS), [Ges[j][1], Ges[j][2], G, t, B]]
      end if
    end for
  end for
  {PF4Basis(SYS[m])}_{m=1}^{|SYS|}
end while
Return (OUTSYS)
```

Theorem 2.4. *The* PF_4 *algorithm terminates in finitely many steps and is correct.*

Proof. To prove the termination of this algorithm we can consider this computation like a tree graph and its node corresponds to a triple which is the output of the GES algorithm. The number of branches is finite (due to the termination of the original F_4 algorithm). Also, the number of nodes in this tree is finite (by termination of the GES algorithm) and all these arguments together conclude the termination of the algorithm.

Morever, the correctness of the algorithm is guaranteed by the correctness of the GES and the F_4 algorithms. Since, we basically follow the structure of the F_4 algorithm, we get at the end a Gröbner system of the input ideal. □

Example 2.5. Let $F = [(c^2-1)x^2+b^2-1, (a^2-1)xy^2+c+b] \subset \mathbb{K}[a,b,c][x,y]$ where x, y are varibles and a, b, c are parameters. We consider the monomial orderings $y \prec_{lex} x$ and $c \prec_{lex} b \prec_{lex} a$. Using our implementation of the PF4 algorithm in Maple, we obtain the following Gröbner system of $\langle F\rangle$ w.r.t. the product ordering $y \prec_{lex} x$ and $c \prec_{lex} b \prec_{lex} a$ saved in OUTSYS as follows:

$$\left\{\begin{array}{ll}
([a^2-1],[b+c,c-1,c+1], & [1]),\\
([c^2-1],[a-1,a+1,b-1,b+1], & [1]),\\
([c^2-1,a^2-1],[b-1,b+1], & [1]),\\
([c^2-1,b^2-1,a^2-1],[b+c], & [1]),\\
([c^2-1,b+c,a^2-1],[\,], & [\,]),\\
([b^2-1],[a-1,a+1,b+c,c-1,c+1,2bc+c^2+1], & [c^2x^2-x^2,a^2xy^2-xy^2+b+c,bc^2x+c^3x-bx-cx,2bc^3+c^4-2bc-1]),\\
([b+c,a^2-1],[c-1,c+1], & [c^2x^2+c^2-x^2-1]),\\
([c^2-1,b^2-1],[a-1,a+1], & [a^2xy^2-xy^2+b+c]),\\
([b+c],[a-1,a+1,c-1,c+1], & [c^2x^2+c^2-x^2-1,a^2xy^2-xy^2,a^2c^2y^2-a^2y^2-c^2y^2+y^2]),\\
([\,],[a-1,a+1,b-1,b+1,b+c,c-1,c+1], & [c^2x^2+b^2-x^2-1,a^2xy^2-xy^2+b+c,a^2b^2y^2-a^2y^2-b^2y^2-\\
 & bc^2x-c^3x+bx+cx+y^2,y^4a^4b^2-y^4a^4-2a^2b^2y^4+2a^2y^4+\\
 & b^2c^2+(y^4-1)b^2+2bc^3-2bc+c^4-c^2-y^4]).
\end{array}\right.$$

In the following, we are willing to compare the performance of the PF_4 algorithm with PGBMain and Improved-DisPGB algorithms (an improvement of the DisPGB algorithm [10] equipped to the Update algorithm [7]). For this purpose, we have implemented all the algorithms described in this paper in Maple 15[2]. In this direction, the following parametric ideals in the ring $\mathcal{S} = \mathbb{Q}[a,b,c,d,m,n,r,t][x,y,z,u,v,w]$ have been chosen, and our aim was to compute a Gröbner system of the ideal generated by each list of polynomials with respect to the product of the orderings $v \prec_{lex} w \prec_{lex} u \prec_{lex} z \prec_{lex} y \prec_{lex} x$ and $t \prec_{lex} r \prec_{lex} n \prec_{lex} m \prec_{lex} d \prec_{lex} c \prec_{lex} b \prec_{lex} a$.

- EX.1= $[ab^4cuxz - a - c, aby^2 - a^2 + b^2, abuxz - c]$
- EX.2= $[(a-c)xz - x, (-b^3 + a^2)uxz - ab, (a+b)y - a^2]$
- EX.3= $[(c^2-1)x^2y + b^2 - 1, (a^2-1)x^2z + c + b, (a-b)y^2z - x - 1, bxy + a - c]$
- EX.4= $[bx^2z^3 - n^3 + n, cx^2y^3 - a^3 - a, dx^3y^2 - m^3 - m]$
- EX.5= $[abcxyz - a - b - c, abxy - a - b, ax^3 - bc, by^3 - c, cz^3 - a]$
- EX.6= $[(bc-1)x^2 + c^2 - a, (ab-1)z^2 - c + a, y^2 - (b-1)x - 1, (b+c+a)z^2 + a + b + c]$
- EX.7= $[(c-a-b)x^3z^3 + c^3 - a - b, (b-a-c)z^2y^5 - c - a - m, (a-m+n)z^2 - a + b]$
- EX.8= $[(a-1)xyz + a, (b-2)y^2 + ab, (c+a)xy - a - 1]$
- EX.9= $[ab^4tux^3 - x - a^3 - n, abxy^3 + b^4 - a^2 + a, anxz^3 + a - 1]$

Example	Method	Time (sec.)	Used Memory (GB)
EX.1	PF4	0.38	0.007
	PGBMain	0.51	0.016
	FirstGB	0.2	0.006
	Improved-DisPGB	0.54	0.018
EX.2	PF4	0.9	0.02
	PGBMain	0.41	0.012
	FirstGB	0.3	0.005
	Improved-DisPGB	1.44	0.035
EX.3	PF4	11.64	0.72
	PGBMain	—	—
	FirstGB	—	—
	Improved-DisPGB	—	—
EX.4	PF4	3.52	0.09
	PGBMain	27.81	1.91
	FirstGB	0.32	0.015
	Improved-DisPGB	6.1	0.23
EX.5	PF4	12.48	1.2
	PGBMain	—	—
	FirstGB	—	—
	Improved-DisPGB	—	—
EX.6	PF4	1.13	0.041
	PGBMain	2.17	0.1
	FirstGB	0.02	0.016
	Improved-DisPGB	1.84	0.048
EX.7	PF4	1.43	0.036
	PGBMain	—	—
	FirstGB	197.25	27.64
	Improved-DisPGB	2.59	0.075
EX.8	PF4	0.67	0.022
	PGBMain	031	0.007
	FirstGB	0.14	0.001
	Improved-DisPGB	1.04	0.029
EX.9	PF4	9.71	0.45
	PGBMain	11.23	0.85
	FirstGB	4.81	0.51
	Improved-DisPGB	50.21	6.3

The results are shown in the comparison table where the timings were conducted on personal computer with 5 core, 4 GB RAM and 64 bits under the Windows 10 operating system. The row "First GB" stands for the computation of the reduced Gröbner basis of the corresponding ideal in the polynomial ring $\mathbb{K}[\mathbf{a},\mathbf{x}]$ with respect to $\prec_{\mathbf{x},\mathbf{a}}$ using the

[2]The Maple codes of the algorithms are available at `http://amirhashemi.iut.ac.ir/softwares` under the names `PF4.mpl`, `PLA-PFGLM.mpl` and `Montes.mpl`. The first file contains a Maple implementation of our algorithm for computing Gröbner systems. The second file contains a Maple implementation of the Kapur et al. algorithm (PGBMain algorithm) and the last one is a Maple implementation of the improvement of the Montes DisPGB algorithm.

MAPLE function `Basis`. Furthermore, the third and fourth columns show respectively the CPU time (in seconds) and the amount of used memory (in gigabytes) of the total computation by the corresponding method. It is worth noting that, this computation is needed the first step in the PGBMAIN algorithm to compute a Gröbner system with respect to $\prec_{\mathbf{x},\mathbf{a}}$. Also, the symbol "–" means that the results can not computed within 600 seconds.

References

[1] BUCHBERGER, B. A criterion for detecting unnecessary reductions in the construction of Gröbner-bases. Symbolic and algebraic computation, EUROSAM '79, int. Symp., Marseille 1979, Lect. Notes Comput. Sci. 72, 3-21 (1979)., 1979.

[2] BUCHBERGER, B. Bruno Buchberger's PhD thesis 1965: An algorithm for finding the basis elements of the residue class ring of a zero dimensional polynomial ideal. Translation from the German. *J. Symb. Comput. 41*, 3-4 (2006), 475–511.

[3] DEHGHANI DARMIAN, M., AND HASHEMI, A. Parametric FGLM algorithm. *J. Symb. Comput. 82* (2017), 38–56.

[4] DEHGHANI DARMIAN, M., HASHEMI, A., AND MONTES, A. Erratum to "A new algorithm for discussing Gröbner bases with parameters" [J. Symbolic Comput. 33 (1-2) (2002) 183-208]. *J. Symb. Comput. 46*, 10 (2011), 1187–1188.

[5] FAUGÈRE, J.-C. A new efficient algorithm for computing Gröbner bases (F_4). *J. Pure Appl. Algebra 139*, 1-3 (1999), 61–88.

[6] FAUGÈRE, J.-C. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *Proceedings of the 2002 international symposium on symbolic and algebraic computation, ISSAC 2002, Lille, France, July 07–10, 2002*. New York, NY: ACM Press, 2002, pp. 75–83.

[7] HASHEMI, A., DEHGHANI DARMIAN, M., AND M.-ALIZADEH, B. Applying Buchberger's criteria on Montes's DisPGB algorithm. *Bull. Iran. Math. Soc. 38*, 3 (2012), 715–724.

[8] KAPUR, D., SUN, Y., AND WANG, D. A new algorithm for computing comprehensive Gröbner systems. In *Proceedings of the 35th international symposium on symbolic and algebraic computation, ISSAC 2010, Munich, Germany, July 25–28, 2010*. New York, NY: Association for Computing Machinery (ACM), 2010, pp. 29–36.

[9] LAZARD, D. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. Computer algebra, EUROCAL '83, Proc. Conf., London 1983, Lect. Notes Comput. Sci. 162, 146-156 (1983)., 1983.

[10] MONTES, A. A new algorithm for discussing Gröbner bases with parameters. *J. Symb. Comput. 33*, 2 (2002), 183–208.

[11] SUZUKI, A., AND SATO, Y. A simple algorithm to compute comprehensive Gröbner bases using Gröbner bases. In *In Proceedings of the 2006 international symposium on symbolic and algebraic computation, ISSAC 06, Genova, Italy, July 9–12, 2006*. New York, NY: ACM Press, 2006, pp. 326–331.

[12] WEISPFENNING, V. Comprehensive Gröbner bases. *J. Symb. Comput. 14*, 1 (1992), 1–29.

[13] WEISPFENNING, V. Canonical comprehensive Gröbner bases. *J. Symb. Comput. 36*, 3-4 (2003), 669–683.

e-mail: m.dehghanidarmian@gmail.com
e-mail: amir.hashemi@ipm.ir