J Syst Sci Complex (2023) 36: 2661–2680

Square-Free Pure Triangular Decomposition of Zero-Dimensional Polynomial Systems*

LI Haokun · XIA Bican · ZHAO Tianqi

DOI: 10.1007/s11424-023-2260-3
Received: 13 June 2022 / Revised: 23 August 2022

Abstract Triangular decomposition with different properties has been used for various types of problem solving. In this paper, the concepts of pure chains and square-free pure triangular decomposition (SFPTD) of zero-dimensional polynomial systems are defined. Because of its good properties, SFPTD may be a key way to many problems related to zero-dimensional polynomial systems. Inspired by the work of Wang (2016) and of Dong and Mou (2019), the authors propose an algorithm for computing SFPTD based on Gröbner bases computation. The novelty of the algorithm is that the authors make use of saturated ideals and separant to ensure that the zero sets of any two pure chains are disjoint and every pure chain is square-free, respectively. On one hand, the authors prove the arithmetic complexity of the new algorithm can be single exponential in the square of the number of variables, which seems to be among the rare complexity analysis results for triangular-decomposition methods. On the other hand, the authors show experimentally that, on a large number of examples in the literature, the new algorithm is far more efficient than a popular triangular-decomposition method based on pseudo-division, and the methods based on SFPTD for real solution isolation and for computing radicals of zero-dimensional ideals are very efficient.

Keywords Gröbner basis, pure chain, square-free pure chain, triangular decomposition, zero-dimensional polynomial system.

1 Introduction

Decomposing a polynomial system to finitely many triangular sets with corresponding zero decomposition, called triangular decomposition of polynomial systems, is one of the fundamental tools in computational ideal theory. Wu first proposed a triangular-decomposition algorithm for computing characteristic sets in [1], which was applied to geometry theorem proving. Since then, triangular decomposition methods have been applied successfully to not only geometry theorem

LI Haokun · XIA Bican · ZHAO Tianqi (Corresponding author)
School of Mathematical Sciences, Peking University, Beijing 100871*, China.*
Email: haokunli@pku.edu.cn; xbc@math.pku.edu.cn; zhaotq@pku.edu.cn.
*This research was supported by National Key R&D Program of China under Grant No. 2022YFA1005102 and the National Natural Science Foundation of China under Grant No. 61732001.
◇ *This paper was recommended for publication by Editor MOU Chenqi.*

proving but also lots of problems with diverse backgrounds, such as automated reasoning, real solution isolation, real solution classification and computing the radical of a polynomial ideal[1–13], to name a few. Different applications may require different types of triangular sets and triangular decomposition with specific properties (see the book [14] for reference). In the passed several decades, many specific triangular sets or triangular systems have been defined, e.g., regular chains, normal chains and square-free triangular sets. And, lots of triangular-decomposition algorithms have been proposed, see for example [4, 15–24]. Nevertheless, classical triangular-decomposition algorithms are mainly based on factorization and pseudo-division and are well known not so efficient on big examples. Another significant but somehow neglected aspect is that, there are few results about the complexity of classical triangular-decomposition algorithms.

The Gröbner basis method, first proposed by Buchberger in [25], has been extensively studied and applied to lots of research fields. It is well known that the complexity of computing Gröbner bases can be double-exponential time in general case and single-exponential time for zero-dimensional ideals[26, 27]. There are some famous algorithms for computing Gröbner bases (see for example [28, 29]) and corresponding tools are available in some computer algebra systems, e.g., `Maple`, `Mathematica` and `Magma`.

Then, one may ask whether there exists a connection between triangular sets and Gröbner bases and whether one can obtain triangular decomposition by Gröbner bases computation. For zero-dimensional polynomial systems, Lazard's work[30] and Möller's work[31] solved the problem, as Lazard gave different algorithms to obtain triangular sets from Gröbner bases with respect to (w.r.t.) the lex ordering or other monomial orderings and Möller proposed a triangular-decomposition algorithm by means of reduced Gröbner bases with respect to the lex ordering (written as reduced LEX Gröbner bases). In general case, the problem was well solved in Wang's work[32] in 2016. The key concept is the W-characteristic set, which is a minimal triangular set extracted from a reduced LEX Gröbner basis. Wang proved that if the variable ordering condition is satisfied for a W-characteristic set, then the regularity and normality of the W-characteristic set are equivalent. Later, Dong and Mou proposed some algorithms in [19, 21] for characteristic decomposition which is also a type of triangular decomposition consisting of normal chains. Owe to efficient computation of Gröbner bases, their algorithms perform better on some complicated polynomial systems than the classical algorithms.

In this paper, we only consider zero-dimensional systems. We define a new type of triangular sets, namely *pure chains* (see Definition 3.1). And then, a new type of triangular decomposition, called square-free pure triangular decomposition (SFPTD) (see Definition 3.3), is introduced for real solution isolation. We observe that SFPTD can also be applied to computing radicals of zero-dimensional ideals. So, the main goal of this paper is to efficiently compute SFPTD of zero-dimensional systems. More formally, we have the following problem statement:

Input: A nonempty finite set $F \subseteq \mathbb{Q}[x_1, \cdots, x_n] \setminus \{0\}$.

Output: An SFPTD $\{\mathcal{T}_1, \cdots, \mathcal{T}_s\}$ of F, where each $\mathcal{T}_i$ is a square-free pure chain, such that

- the zero set of F equals the union of all zero sets of $\mathcal{T}_i$, and

- any two zero sets of $\mathcal{T}_i$ and $\mathcal{T}_j$ $(i \neq j)$ have no intersection,

if F is zero-dimensional; FAIL otherwise.

We list our main contributions as follows:

1) Inspired by [19, Algorithm 1], we propose an algorithm (Algorithm 1) for computing SFPTD.

2) We prove that the arithmetic complexity of Algorithm 1 can be single exponential in the square of the number of variables.

3) We implemented Algorithm 1 with `Maple2021`. Our experiments show that Algorithm 1 is far more efficient than the classical method for zs-rc decomposition in [9] (see the group of columns TD in Table 2).

4) By the methods based on SFPTD, one can compute isolating cubes of real solutions to the system, and the radical of the ideal generated by the system, efficiently (see the groups of columns RSI and RA in Table 2).

The rest of this paper is organized as follows. In Section 2, we recall some basic concepts and existing results about the theories of Gröbner bases and triangular sets. In Section 3, we define pure chains and propose Algorithm 1 to compute SFPTD of zero-dimensional systems. The termination and correctness of Algorithm 1 is guaranteed by Theorem 3.11. In Section 4, we analyze the arithmetic complexity of Algorithm 1. In Section 5, we present two applications of SFPTD: Real solution isolation and computing the radical of a zero-dimensional ideal. In Section 6, we explain the implementation details and show the experimental results. Section 7 concludes the paper.

2 Preliminary

In the section, we recall some basic concepts and existing results about the theories of Gröbner bases and triangular sets. The reader is referred to [14, 33] for more details.

2.1 Zero-Dimensional Systems

Let $x_1, \cdots, x_n$ be n variables and let $\overline{\boldsymbol{x}}$ denote the vector $(x_1, \cdots, x_n)$. Throughout the paper, we fix the variable ordering $x_1 < \cdots < x_n$.

Denote by $\mathbb{Q}$ and $\mathbb{C}$ the set of rational and complex numbers, respectively. For any $\mathbb{A} \in \{\mathbb{Q}, \mathbb{C}\}$, we denote by $\mathbb{A}[\overline{\boldsymbol{x}}]$ or $\mathbb{A}[x_1, \cdots, x_n]$ the corresponding polynomial ring generated by the variables $x_1, \cdots, x_n$. For any $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$, we denote by $\langle F \rangle$ *the ideal* generated by F in $\mathbb{C}[\overline{\boldsymbol{x}}]$. For any ideal $I \subseteq \mathbb{C}[\overline{\boldsymbol{x}}]$, $\mathtt{V}(I)$ denotes the *affine variety* $\{(a_1, \cdots, a_n) \in \mathbb{C}^n \mid f(a_1, \cdots, a_n) = 0 \text{ for all } f \in I\}$. In particular, $\mathtt{V}(F) := \mathtt{V}(\langle F \rangle)$, where $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$. Let $f \in \mathbb{C}[\overline{\boldsymbol{x}}]$. We denote the total degree of f and the degree of f with respect to x_i by $\deg(f)$ and $\deg(f, x_i)$, respectively. Fix an admissible monomial ordering. Denote by $\mathtt{LT}(f)$ and $\mathtt{LM}(f)$ the *leading term* and the *leading monomial* of f, respectively.

Definition 2.1 For $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$, F is a *zero-dimensional system* or $\langle F \rangle$ is a *zero-dimensional ideal*, if $\mathtt{V}(F)$ is a finite set.

Definition 2.2 Fix a monomial ordering and let $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$. A finite subset $G = \{g_1, \cdots, g_t\} \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$ is called a *Gröbner basis* of $\langle F\rangle$ if $\langle \mathtt{LT}(g_1), \cdots, \mathtt{LT}(g_t)\rangle = \langle \mathtt{LT}(\langle F\rangle)\rangle$, where $\mathtt{LT}(\langle F\rangle) = \{\mathtt{LT}(f) \mid f \in \langle F\rangle\}$.

Proposition 2.3 ([33, Chap. 8.3]) *Let $F \subseteq \mathbb{Q}[x_1, \cdots, x_n]$. The following statements are equivalent:*

(a) *F is a zero-dimensional system,*

(b) *for every Gröbner basis G of $\langle F\rangle$, G contains n polynomials $g_1, \cdots, g_n$ such that $\mathtt{LM}(g_i) = x_i^{k_i}$ $(k_i \geq 1)$,*

(c) *there exist a monomial ordering $\prec$ and a Gröbner basis of $\langle F\rangle$ with respect to $\prec$ which contains n polynomials $g_1, \cdots, g_n$ such that $\mathtt{LM}(g_i) = x_i^{k_i}$ $(k_i \geq 1)$.*

2.2 Triangular Sets

For $f \in \mathbb{Q}[\overline{\boldsymbol{x}}] \setminus \mathbb{Q}$, we denote by $\mathtt{lv}(f)$ the *main variable* of f and by $\mathtt{ini}(f)$ the *initial* (or leading coefficient with respect to $\mathtt{lv}(f)$) of f. For $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$, $\mathtt{lv}(F) := \{\mathtt{lv}(f) \mid f \in F\}$ and $\mathtt{ini}(F) := \{\mathtt{ini}(f) \mid f \in F\}$.

Definition 2.4 Let $\mathcal{T} = [T_1, \cdots, T_t]$ be a finite nonempty list of nonconstant polynomials in $\mathbb{Q}[\overline{\boldsymbol{x}}]$. We call $\mathcal{T}$ a *triangular set* if $\mathtt{lv}(T_1) < \cdots < \mathtt{lv}(T_t)$.

Let $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$, f_1 and f_2 be two polynomials in $\mathbb{Q}[\overline{\boldsymbol{x}}]$, and $\mathcal{T} = [T_1, \cdots, T_t]$ be a triangular set in $\mathbb{Q}[\overline{\boldsymbol{x}}]$. The *saturated ideal* of F with respect to f_1 is defined as $\langle F\rangle : f_1^{\infty} := \{g \in \mathbb{C}[\overline{\boldsymbol{x}}] \mid \text{there exists } i \geq 0 \text{ such that } f_1^i g \in \langle F\rangle\}$. And, we denote by $\mathtt{sat}(\mathcal{T})$ the saturated ideal of $\mathcal{T}$, namely $\langle \mathcal{T}\rangle : (\mathtt{ini}(T_1) \cdots \mathtt{ini}(T_t))^{\infty}$. The *resultant* of f_1 and f_2 with respect to $\mathtt{lv}(f_2)$ is denoted by $\mathtt{res}(f_1, f_2, \mathtt{lv}(f_2))$, and the resultant of f and $\mathcal{T}$ is defined as $\mathtt{res}(f, \mathcal{T}) := \mathtt{res}(\cdots \mathtt{res}(\mathtt{res}(f, T_t, \mathtt{lv}(T_t)), T_{t-1}, \mathtt{lv}(T_{t-1})), \cdots, T_1, \mathtt{lv}(T_1))$.

Definition 2.5 Let $\mathcal{T} = [T_1, \cdots, T_t] \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$ be a triangular set. The triangular set is called a *regular chain* or is said to be *regular*, if $\mathtt{ini}(T_1) \neq 0$ and for each i $(2 \leq i \leq t)$, $\mathtt{res}(\mathtt{ini}(T_i), [T_1, \cdots, T_{i-1}]) \neq 0$. The triangular set is called a *normal chain* or is said to be *normal*, if $\mathtt{ini}(\mathcal{T})$ does not involve the main variables of $\mathcal{T}$.

Remark 2.6 It is clear that any normal chain is a regular chain.

Definition 2.7 For any $f \in \mathbb{Q}[\overline{\boldsymbol{x}}] \setminus \mathbb{Q}$, we denote by $\mathtt{sep}(f)$ the *separant* of f, i.e., $\partial f / \partial\, \mathtt{lv}(f)$. A triangular set $\mathcal{T} = [T_1, \cdots, T_t]$ in $\mathbb{Q}[\overline{\boldsymbol{x}}]$ is said to be *square-free*, if the discriminant of T_1 with respect to $\mathtt{lv}(T_1)$ is not equal to 0, and for each i $(2 \leq i \leq t)$, $\mathtt{res}(\mathtt{sep}(T_i), [T_1, \cdots, T_i]) \neq 0$.

2.3 *W*-Characteristic Sets

Definition 2.8 ([32, Def. 3.1]) Let $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$ and $G \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$ be the reduced Gröbner basis of $\langle F\rangle$ with respect to the lex ordering $\prec_{\mathtt{lex}}$. Define $G_i := \{g \in G \mid \mathtt{lv}(g) = x_i\}$ for $i = 1, \cdots, n$. In every nonempty G_i, there exists a unique polynomial g_i such that $\mathtt{LM}(g_i) \prec_{\mathtt{lex}} \mathtt{LM}(g)$ for any $g \in G_i \setminus \{g_i\}$. The ordered list of all g_i is called the *W-characteristic set* of F.

Recall that we fix the variable ordering $x_1 < \cdots < x_n$. We say that *the variable ordering condition* is satisfied for a W-characteristic set $\mathcal{C}$, if the variables in $\{x_1, \cdots, x_n\} \setminus \mathtt{lv}(\mathcal{C})$ are

ordered before $\mathtt{lv}(\mathcal{C})$. The following theorem presents two properties of W-characteristic sets.

Theorem 2.9 ([32, Prop. 3.1 & Thm. 3.9]) *Let $\mathcal{C} = [C_1, \cdots, C_t]$ be the W-characteristic set of $F \subseteq \mathbb{Q}[x_1, \cdots, x_n]$, where $1 \leq t \leq n$. We have*

(a) $\langle \mathcal{C} \rangle \subseteq \langle F \rangle \subseteq \mathtt{sat}(\mathcal{C})$, *and*

(b) *if the variable ordering condition is satisfied for $\mathcal{C}$ and $\mathcal{C}$ is not a normal chain, then there exists an integer k $(1 \leq k < t)$ such that $[C_1, \cdots, C_k]$ is normal and $[C_1, \cdots, C_{k+1}]$ is not regular.*

3 Square-Free Pure Triangular Decomposition

For real solution isolation, we need to compute a finite number of square-free regular chains whose zero sets are pairwise disjoint. In fact, we can compute some triangular sets stronger than regular chains and we define them as *pure chains* in Subsection 3.1. For computing square-free/regular chains, a popular method is the method of relatively simplicial decomposition (see [12, Chapter 2] for more details), which is based on subresultant computation and pseudo-division. Different to the method, we propose an algorithm to compute square-free pure triangular decomposition by means of Gröbner bases in Subsection 3.2. We explain two sub-algorithms of the algorithm in Subsection 3.3 and Subsection 3.4, respectively.

3.1 Pure Chains

Definition 3.1 Let $\mathcal{T} = [T_1, \cdots, T_n] \subseteq \mathbb{Q}[x_1, \cdots, x_n]$ be a triangular set. The triangular set is called a *pure chain* or is said to be *pure*, if $\mathtt{ini}(\mathcal{T}) \subseteq \mathbb{Q} \setminus \{0\}$ and $\mathtt{lv}(T_i) = x_i$ for $i = 1, \cdots, n$. A pure chain is said to be *reduced*, if for each i $(1 \leq i \leq n)$, $\mathtt{ini}(T_i) = 1$ and $\deg(T_i, x_i) > \deg(T_j, x_i)$ where $j > i$.

Obviously, any pure chain is a normal chain. And, pure chains have very good properties.

Proposition 3.2 *Fix the lex monomial ordering and let $\mathcal{T} = [T_1, \cdots, T_n] \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$ be a pure chain. Then,*

(a) $\mathtt{sat}(\mathcal{T}) = \langle \mathcal{T} \rangle$,

(b) $\mathtt{LM}(T_i) = x_i^{j_i}$ $(j_i \geq 1)$ *for* $i = 1, \cdots, n$,

(c) $\mathcal{T}$ *is an LEX Gröbner basis, and*

(d) $\mathcal{T}$ *is a zero-dimensional system.*

Furthermore, $\mathcal{T}$ is reduced if and only if $\mathcal{T}$ is a reduced LEX Gröbner basis.

Proof (a) It is because $\mathtt{ini}(\mathcal{T}) \subseteq \mathbb{Q} \setminus \{0\}$. (b) It is clear. (c) By (b), for any $i_1 \neq i_2$, the greatest common divisor of $\mathtt{LM}(T_{i_1})$ and $\mathtt{LM}(T_{i_2})$ is 1. Then, by [33, Lemma 5.66], we complete the proof. (d) It is obvious by (b), (c) and Proposition 2.3. ∎

Definition 3.3 Let $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$ be a zero-dimensional system. A *square-free pure triangular decomposition* (SFPTD) of F is a finite set of square-free pure chains $\{\mathcal{T}_1, \cdots, \mathcal{T}_s\}$, where $\mathcal{T}_i \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$, such that $\mathtt{V}(F) = \bigcup_{i=1}^{s} \mathtt{V}(\mathcal{T}_i)$ and $\mathtt{V}(\mathcal{T}_i) \cap \mathtt{V}(\mathcal{T}_j) = \emptyset$ for any $i \neq j$.

3.2 The Two-Step Algorithm

For any zero-dimensional system $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$, we propose Algorithm 1 to compute an SFPTD of F. The algorithm has the following two steps.

Step 1 Decompose F to a finite number of pure chains such that their zero sets are pairwise disjoint. There are many existing methods to complete this step, like the method in [31]. In order to analyze the complexity of Algorithm 1, we present an algorithm named **PureDec** here. The algorithm is very similar to [19, Algorithm 1] in a zero-dimensional case and the details will be given in Algorithm 2.

Algorithm 1 SfPureDec

Input : a nonempty finite set $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}] \setminus \{0\}$ and the vector $\overline{\boldsymbol{x}}$
Output: $ans = \{\mathcal{T}_1, \cdots, \mathcal{T}_s\}$, a finite set of square-free pure chains such that

$$\mathtt{V}(F) = \bigcup_{\mathcal{T}_i \in ans} \mathtt{V}(\mathcal{T}_i) \text{ and } \mathtt{V}(\mathcal{T}_i) \cap \mathtt{V}(\mathcal{T}_j) = \emptyset \text{ for any } i \neq j,$$

if F is zero-dimensional; FAIL otherwise

```
Λ ← PureDec(F, x̄)
if Λ = FAIL then
    return FAIL
else
    return ⋃_{P∈Λ} SubSfPureDec(P, x̄)
```

Algorithm 2 PureDec (Sub-Algorithm of Algorithm 1)

Input : a nonempty finite set $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}] \setminus \{0\}$ and the vector $\overline{\boldsymbol{x}}$
Output: $ans = \{\mathcal{T}_1, \cdots, \mathcal{T}_s\}$, a finite set of pure chains such that

$$\mathtt{V}(F) = \bigcup_{\mathcal{T}_i \in ans} \mathtt{V}(\mathcal{T}_i) \text{ and } \mathtt{V}(\mathcal{T}_i) \cap \mathtt{V}(\mathcal{T}_j) = \emptyset \text{ for any } i \neq j,$$

if F is zero-dimensional; FAIL otherwise

```
ans ← ∅, Φ ← {F}, num ← 0
while Φ ≠ ∅ do
    num ← num + 1
    Choose P from Φ and set Φ ← Φ \ {P}
    G ← the reduced LEX Gröbner basis of ⟨P⟩
    if G ≠ {1} then
        if num = 1 and there exists x_i such that for any g ∈ G, LM(g) ≠ x_i^k (k ≥ 1) then
            return FAIL
        C ← [C_1, ··· , C_m] (m ≤ n) which is the W-characteristic set of G
        # In fact, we have m = n (see the termination proof of Theorem 3.7).
        if C is a pure chain then
            ans ← ans ∪ {C}
        else
            C_k ← the first polynomial of C that makes [C_1, ··· , C_k] not pure
            G_sat ← the reduced Gröbner basis of ⟨C_1, ··· , C_{k−1}⟩ : ini(C_k)^∞ w.r.t. any monomial ordering
            Φ ← Φ ∪ {G ∪ {ini(C_k)}} ∪ {G ∪ G_sat}
return ans
```

Step 2 Compute an SFPTD of every pure chain in **Step 1**. The step is done by our proposed algorithm, named **SubSfPureDec**. The details will be given in Algorithm 3 in Subsection 3.4. It is clear that the union of all SFPTD is an SFPTD of F.

Remark that if the input system is not zero-dimensional, Algorithm 1 returns FAIL in Line 3.

3.3 Step 1 Decomposing to Pure Chains

Since the algorithm **PureDec** is similar to [19, Algorithm 1], we do not explain it step by step and only give the pseudocode in Algorithm 2.

Remark 3.4 Except for the following two aspects, Algorithm 2 is essentially the same as [19, Algorithm 1] in a zero-dimensional case. In order to guarantee the zero sets to be pairwise disjoint, Algorithm 2 computes $\langle C_1, \cdots, C_{k-1}\rangle : \mathtt{ini}(C_k)^\infty$ in Line 15 instead of the ideal quotient of $\langle C_1, \cdots, C_{k-1}\rangle$ by $\mathtt{ini}(C_k)$ (see [19, Algorithm 1-Line 19]). And Algorithm 2 can detect whether the input system is zero-dimensional.

The correctness and termination of [19, Algorithm 1] is guaranteed by [19, Thm. 2]. Naturally, we can show the correctness and termination of Algorithm 2 in a similar way. However, for the sake of our complexity analysis in Section 4, we prepare two lemmas and give a complete proof.

Lemma 3.5 *Let $\mathcal{C}$ be the W-characteristic set of $F \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$. If $\mathcal{C}$ is a pure chain, then $\langle F\rangle = \langle \mathcal{C}\rangle$.*

Proof It is clear by Proposition 3.2 (a) and Theorem 2.9 (a). ∎

For any affine variety $V \subseteq \mathbb{C}^n$, define *the radical ideal* $\mathtt{I}(V) := \{f \in \mathbb{C}[\overline{\boldsymbol{x}}] \mid f(a_1, \cdots, a_n) = 0 \text{ for any } (a_1, \cdots, a_n) \in V\}$.

Lemma 3.6 *Let $\mathcal{T} \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$ be a pure chain and $f \in \mathbb{Q}[\overline{\boldsymbol{x}}]$. If $\mathtt{res}(f, \mathcal{T}) = 0$, there exists $g \in \mathbb{C}[\overline{\boldsymbol{x}}] \setminus \langle \mathcal{T}\rangle$ such that $fg \in \langle \mathcal{T}\rangle$.*

Proof Let $V_1 := \mathtt{V}(\mathcal{T})$ and $V_2 := \mathtt{V}(\mathcal{T}) \setminus \mathtt{V}(f)$. Since $\mathcal{T}$ is a zero-dimensional system by Proposition 3.2 (d), V_2 is an affine variety. By [12, Theorem 2.2], $\mathtt{res}(f, \mathcal{T}) = 0$ if and only if $\mathtt{V}(\mathcal{T}) \cap \mathtt{V}(f) \neq \emptyset$, i.e., $\mathtt{V}_1 \neq \emptyset$ and V_2 is a proper subset of V_1. Then, it is equivalent to that there exists $q \in \mathtt{I}(V_2) \setminus \mathtt{I}(V_1)$. Note that $fq \in \mathtt{I}(V_1)$. Then, there exists some integer $k_1 \geq 1$ such that $(fq)^{k_1} \in \langle \mathcal{T}\rangle$. Let k_2 be the smallest integer number such that $f^{k_2}q^{k_1} \in \langle \mathcal{T}\rangle$. Since $q \notin \mathtt{I}(V_1)$, $k_2 \geq 1$. Thus, we take $g = f^{k_2-1}q^{k_1}$ which is not in $\langle \mathcal{T}\rangle$. ∎

Theorem 3.7 *Algorithm* 2 *terminates correctly.*

Proof (Correctness) For the input polynomial set F, let G_0 be the reduced LEX Gröbner basis of $\langle F\rangle$. By Proposition 2.3, if there exists x_i such that for any $g \in G_0$, $\mathtt{LM}(g) \neq x_i^k$ $(k \geq 1)$, then F is not zero-dimensional. Algorithm 2 returns FAIL in Line 8.

Otherwise, F is zero-dimensional. In every loop, we pick a polynomial set P from Φ. Let G $(G \neq \{1\})$ be the reduced LEX Gröbner basis of $\langle P\rangle$ and $\mathcal{C} = [C_1, \cdots, C_m]$ $(m \leq n)$ be the W-characteristic set of G. If $\mathcal{C}$ is pure, then we add $\mathcal{C}$ to the output set in Line 12. If $\mathcal{C}$ is not pure, Φ is updated with two sets in Line 16. Therefore, we only need to prove that

(I) $\mathtt{V}(G) = \mathtt{V}(\mathcal{C})$, where $\mathcal{C}$ is pure,

(II) $\mathtt{V}(G\cup\{\mathtt{ini}(C_k)\})\cap\mathtt{V}(G\cup G_{sat})=\emptyset$ and $\mathtt{V}(G)=\mathtt{V}(G\cup\{\mathtt{ini}(C_k)\})\cup\mathtt{V}(G\cup G_{sat})$, where C_k is the first polynomial that makes $[C_1,\cdots,C_k]$ not pure and G_{sat} is a Gröbner basis of $\langle C_1,\cdots,C_{k-1}\rangle:\mathtt{ini}(C_k)^\infty$.

By Lemma 3.5, (I) is clear. It remains to prove (II). Since $[C_1,\cdots,C_{k-1}]$ is pure, by Proposition 3.2 (d), $\{C_1,\cdots,C_{k-1}\}$ is zero-dimensional. Then, by [34, Chap. 4.4, Thm. 10], we have $\mathtt{V}(\langle C_1,\cdots,C_{k-1}\rangle:\mathtt{ini}(C_k)^\infty)=\mathtt{V}(C_1,\cdots,C_{k-1})\setminus\mathtt{V}(\mathtt{ini}(C_k))$. Hence, $\mathtt{V}(G\cup G_{sat})=\mathtt{V}(G)\cap(\mathtt{V}(C_1,\cdots,C_{k-1})\setminus\mathtt{V}(\mathtt{ini}(C_k)))$. Note that $\mathtt{V}(G)\subseteq\mathtt{V}(C_1,\cdots,C_{k-1})$. So, $\mathtt{V}(G\cup G_{sat})=\mathtt{V}(G)\setminus\mathtt{V}(\mathtt{ini}(C_k))$. Then, because $\mathtt{V}(G\cup\{\mathtt{ini}(C_k)\})=\mathtt{V}(G)\cap\mathtt{V}(\mathtt{ini}(C_k))$, the proof is completed.

(Termination) If F is not zero-dimensional, the termination is obvious. Otherwise, the termination is equivalent to that every ideal generated by the added set in Line 16 is strictly larger than that generated by the removed one. So, we only need to prove that $\langle G\cup\{\mathtt{ini}(C_k)\}\rangle$ and $\langle G\cup G_{sat}\rangle$ are both strictly larger than $\langle G\rangle$. Because G is a reduced Gröbner basis, we have $\mathtt{ini}(C_k)\notin\langle G\rangle$. Then, $\langle G\cup\{\mathtt{ini}(C_k)\}\rangle$ is strictly larger than $\langle G\rangle$. It remains to prove $\langle G\rangle\subsetneq\langle G\cup G_{sat}\rangle$.

Firstly, we prove that $\mathcal{C}=[C_1,\cdots,C_n]$ with $\mathtt{lv}(C_i)=x_i$, i.e., $\mathtt{lv}(\mathcal{C})=\{x_1,\cdots,x_n\}$. Since the input F is zero-dimensional, by the proof of the correctness, every polynomial set in Φ is also zero-dimensional. Then, by Proposition 2.3 and Definition 2.8, we have $\mathtt{lv}(\mathcal{C})=\{x_1,\cdots,x_n\}$.

Secondly, we prove that $\mathtt{res}(\mathtt{ini}(C_k),[C_1,\cdots,C_{k-1}])=0$. Note that C_k is the first polynomial that makes $[C_1,\cdots,C_k]$ not pure. Then, $[C_1,\cdots,C_k]$ and $\mathcal{C}$ are not normal, but $[C_1,\cdots,C_{k-1}]$ is normal. Since $\mathtt{lv}(\mathcal{C})=\{x_1,\cdots,x_n\}$, the variable ordering condition is satisfied for $\mathcal{C}$. So, by Theorem 2.9 (b), $[C_1,\cdots,C_k]$ is not regular. Then, $\mathtt{res}(\mathtt{ini}(C_k),[C_1,\cdots,C_{k-1}])=0$.

Finally, we prove $\langle G\rangle\subsetneq\langle G\cup G_{sat}\rangle$, which is equivalent to proving that there exists $g\in\langle G_{sat}\rangle\setminus\langle G\rangle$. Note that $[C_1,\cdots,C_{k-1}]$ is a pure chain in $\mathbb{Q}[x_1,\cdots,x_{k-1}]$. So, by the conclusion in the above paragraph and by Lemma 3.6, there exists $g\in\mathbb{C}[x_1,\cdots,x_{k-1}]\setminus\langle C_1,\cdots,C_{k-1}\rangle$ such that $g\cdot\mathtt{ini}(C_k)\in\langle C_1,\cdots,C_{k-1}\rangle$. Recall that $\langle G_{sat}\rangle=\langle C_1,\cdots,C_{k-1}\rangle:\mathtt{ini}(C_k)^\infty$. So, we have $g\in\langle G_{sat}\rangle$. Then, we only need to prove $g\notin\langle G\rangle$. Note that $\langle G\rangle\cap\mathbb{C}[x_1,\cdots,x_{k-1}]=\langle G\cap\mathbb{C}[x_1,\cdots,x_{k-1}]\rangle$ and $[C_1,\cdots,C_{k-1}]$ is the W-characteristic set of $G\cap\mathbb{C}[x_1,\cdots,x_{k-1}]$. Thus, by Lemma 3.5, $\langle G\rangle\cap\mathbb{C}[x_1,\cdots,x_{k-1}]=\langle C_1,\cdots,C_{k-1}\rangle$. Then, because $g\in\mathbb{C}[x_1,\cdots,x_{k-1}]\setminus\langle C_1,\cdots,C_{k-1}\rangle$, $g\notin\langle G\rangle$. ∎

Corollary 3.8 *If a zero-dimensional system $F\subseteq\mathbb{Q}[\overline{\boldsymbol{x}}]$ is decomposed to a finite set of pure chains $\{\mathcal{T}_1,\cdots,\mathcal{T}_s\}$ by Algorithm 2, then for each i, $\mathcal{T}_i$ is reduced and $\langle F\rangle\subseteq\langle\mathcal{T}_i\rangle$.*

Proof It is clear by the proof of Theorem 3.7, by Proposition 3.2 and by Lemma 3.5. ∎

3.4 Step 2: Computing SFPTD of Pure Chains

The algorithm **SubSfPureDec** (Algorithm 3) computes an SFPTD of any pure chain $\mathcal{T}\subseteq\mathbb{Q}[x_1,\cdots,x_n]$. Let Φ be a set of pure chains for SFPTD (initialized as $\{\mathcal{T}\}$), and *ans* be a set of computed square-free pure chains (initialized as $\emptyset$). Every loop step, we pick a pure chain $\mathcal{P}=[P_1,\cdots,P_n]$ from Φ and remove it from Φ, until Φ is empty.

1) If $\mathcal{P}$ is square-free, then we add $\mathcal{P}$ to *ans*.

2) If $\mathcal{P}$ is not square-free, then we compute k ($k \geq 1$) which is the smallest integer such that $[P_1, \cdots, P_k]$ is not square-free. If $k = 1$, then Φ is updated with the pure chains $[\xi_1, P_2, \cdots, P_n], \cdots, [\xi_m, P_2, \cdots, P_n]$, where ξ_i is an irreducible factor of P_1. If $k > 1$, we compute the reduced Gröbner basis G_{sat} of $\langle P_1, \cdots, P_k\rangle : \mathtt{sep}(P_k)^\infty$ with respect to any monomial ordering. Then, we decompose $\mathcal{P} \cup \{\mathtt{sep}(P_k)\}$ and $\mathcal{P} \cup G_{sat}$ into two finite sets of pure chains by Algorithm 2, respectively. And, Φ is updated with pure chains in the two sets.

Algorithm 3 is illustrated on the following example.

Algorithm 3 SubSfPureDec (Sub-Algorithm of Algorithm 1)

Input : a pure chain $\mathcal{T} = [T_1, \cdots, T_n] \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$ and the vector $\overline{\boldsymbol{x}}$

Output: $ans = \{\mathcal{Q}_1, \cdots, \mathcal{Q}_s\}$, a finite set of square-free pure chains such that

$$\mathtt{V}(\mathcal{T}) = \bigcup_{\mathcal{Q}_i \in ans} \mathtt{V}(\mathcal{Q}_i) \text{ and } \mathtt{V}(\mathcal{Q}_i) \cap \mathtt{V}(\mathcal{Q}_j) = \emptyset \text{ for any } i \neq j$$

```
1  ans ← ∅, Φ ← {T}
2  while Φ ≠ ∅ do
3      Choose P = [P_1, ··· , P_n] from Φ and set Φ ← Φ \ {P}
4      if P is square-free then
5          ans ← ans ∪ {P}
6      else
7          P_k ← the first polynomial that makes [P_1, ··· , P_k] not square-free
8          if k = 1 then
9              ξ_1, ··· , ξ_m ← all irreducible factors of P_1
10             # Here, ξ_i ∈ Q[x̄] for i = 1, ··· , m.
11             Φ ← Φ ∪ {[ξ_1, P_2, ··· , P_n], ··· , [ξ_m, P_2, ··· , P_n]}
12         else
13             G_sat ← the reduced Gröbner basis of ⟨P_1, ··· , P_k⟩ : sep(P_k)^∞ w.r.t. any monomial ordering
14             Φ ← Φ ∪ PureDec(P ∪ {sep(P_k)}, x̄) ∪ PureDec(P ∪ G_sat, x̄)
15 return ans
```

Example 3.9 Consider the pure chain $\mathcal{T} = [T_1, T_2] = [x^2 - x, y^2 - 2xy + 1] \subseteq \mathbb{Q}[x, y]$ with $x < y$. Note that $\mathcal{T}$ is not square-free and T_2 is the first polynomial that makes it not square-free. Since $\mathtt{sep}(T_2) = 2y - 2x$ and the reduced LEX Gröbner basis of $\langle T_1, T_2\rangle : \mathtt{sep}(T_2)^\infty$ is $\{x, y^2+1\}$, we decompose $\mathcal{T} \cup \{\mathtt{sep}(T_2)\}$ into $\{[x-1, y-1]\}$ and $\mathcal{T} \cup \{x, y^2+1\}$ into $\{[x, y^2+1]\}$ by Algorithm 2, respectively. Then, Φ is updated with two pure chains $[x - 1, y - 1]$ and $[x, y^2 + 1]$. Because both pure chains are square-free, Algorithm 3 terminates with $\Phi = \emptyset$ and $\{[x - 1, y - 1],\ [x, y^2 + 1]\}$ is an SFPTD of $\mathcal{T}$.

Theorem 3.10 *Algorithm* 3 *terminates correctly.*

Proof (Correctness) We choose a pure chain $\mathcal{P}$ from Φ in every loop. If $\mathcal{P}$ is square-free, then it is added to the output set in Line 5. Otherwise, Φ is updated with some pure chains in Line 11 or Line 14 (note that since $\mathcal{P}$ is zero-dimensional by Proposition 3.2 (d), Algorithm 2 does not return FAIL in Line 14). So, it is obvious that we only need to prove that $\mathtt{V}(\mathcal{P}) = \mathtt{V}(\mathcal{P} \cup \{\mathtt{sep}(P_k)\}) \cup \mathtt{V}(\mathcal{P} \cup G_{sat})$ and $\mathtt{V}(\mathcal{P} \cup \{\mathtt{sep}(P_k)\}) \cap \mathtt{V}(\mathcal{P} \cup G_{sat}) = \emptyset$, where P_k ($k > 1$) is the first polynomial that makes $[P_1, \cdots, P_k]$ not square-free and G_{sat} is a Gröbner basis of $\langle P_1, \cdots, P_k\rangle : (\mathtt{sep}(P_k))^\infty$. Note that $\{P_1, \cdots, P_k\}$ is also zero-dimensional by Proposition 3.2

(d). Then, similar to the correctness proof of Theorem 3.7, we have $\mathtt{V}(\mathcal{P} \cup G_{sat}) = \mathtt{V}(\mathcal{P}) \setminus \mathtt{V}(\mathtt{sep}(P_k))$ and $\mathtt{V}(\mathcal{P} \cup \{\mathtt{sep}(P_k)\}) = \mathtt{V}(\mathcal{P}) \cap \mathtt{V}(\mathtt{sep}(P_k))$.

(Termination) The termination is equivalent to that every ideal generated by the added pure chain in Line 11 or Line 14 is strictly larger than that generated by the removed one. Note that it is clear for every pure chain added in Line 11. Thus, by Corollary 3.8, we only need to prove that $\langle \mathcal{P} \cup \{\mathtt{sep}(P_k)\}\rangle$ and $\langle \mathcal{P} \cup G_{sat}\rangle$ are both strictly larger than $\langle \mathcal{P}\rangle$.

Firstly, we prove $\langle \mathcal{P}\rangle \subsetneq \langle \mathcal{P} \cup \{\mathtt{sep}(P_k)\}\rangle$. Since $\mathcal{P} = [P_1, \cdots, P_n]$ is a pure chain, by Proposition 3.2, $\mathcal{P}$ is an LEX Gröbner basis, and $\mathtt{LT}(P_i) = c_i x_i^{j_i}$ where $c_i \in \mathbb{Q} \setminus \{0\}$ and $j_i \geq 1$. Then, $\mathtt{LT}(\langle \mathcal{P}\rangle) = \langle \mathtt{LT}(\mathcal{P})\rangle = \langle c_1 x_1^{j_1}, \cdots, c_n x_n^{j_n}\rangle$, and either $\mathtt{LT}(\mathtt{sep}(P_k)) = j_k c_k x_k^{j_k - 1}$ $(j_k > 1)$ or $\mathtt{LT}(\mathtt{sep}(P_k)) = c_k$ $(j_k = 1)$. So, $\mathtt{LT}(\mathtt{sep}(P_k)) \notin \mathtt{LT}(\langle \mathcal{P}\rangle)$. Thus, $\mathtt{sep}(P_k) \notin \langle \mathcal{P}\rangle$. The proof is completed.

Secondly, we prove $\langle \mathcal{P}\rangle \subsetneq \langle \mathcal{P} \cup G_{sat}\rangle$, which is equivalent to proving that there exists $g \in \langle G_{sat}\rangle \backslash \langle \mathcal{P}\rangle$. Note that $[P_1, \cdots, P_{k-1}]$ is square-free, but $[P_1, \cdots, P_k]$ is not. So, $\mathtt{res}(\mathtt{sep}(P_k), [P_1, \cdots, P_k]) = 0$. Then, because $[P_1, \cdots, P_k]$ is a pure chain in $\mathbb{Q}[x_1, \cdots, x_k]$, by Lemma 3.6, there exists $g \in \mathbb{C}[x_1, \cdots, x_k] \setminus \langle P_1, \cdots, P_k\rangle$ such that $g \cdot \mathtt{sep}(P_k) \in \langle P_1, \cdots, P_k\rangle$. Recall that $\langle G_{sat}\rangle = \langle P_1, \cdots, P_k\rangle : (\mathtt{sep}(P_k))^{\infty}$. So, $g \in \langle G_{sat}\rangle$. It remains to prove $g \notin \langle \mathcal{P}\rangle$. Because $\mathcal{P}$ is an LEX Gröbner basis, we have $\langle \mathcal{P}\rangle \cap \mathbb{C}[x_1, \cdots, x_k] = \langle P_1, \cdots, P_k\rangle$. Note that $g \in \mathbb{C}[x_1, \cdots, x_k] \setminus \langle P_1, \cdots, P_k\rangle$. So, $g \notin \langle \mathcal{P}\rangle$. ∎

Theorem 3.11 *Algorithm* 1 *terminates correctly.*

Proof It is obvious by Theorem 3.7 and Theorem 3.10. ∎

4 Arithmetic Complexity Analysis

In the section, we analyze the complexity of our algorithms. Here, we only consider arithmetic complexity which counts the number of field operations (not bit operations). We first introduce the concept of multiplicity and some results we will use later.

For any point $p = (a_1, \cdots, a_n) \in \mathbb{C}^n$, we denote by $\mathbb{C}[\overline{\boldsymbol{x}}]_p$ the set

$$\left\{ \frac{f}{g} \mid f, g \in \mathbb{C}[\overline{\boldsymbol{x}}], g(a_1, \cdots, a_n) \neq 0 \right\}.$$

For any ideal $I \subseteq \mathbb{C}[\overline{\boldsymbol{x}}]$, we denote by $\mathfrak{D}(I)$ the dimension of the $\mathbb{C}$-vector space $\mathbb{C}[\overline{\boldsymbol{x}}]/I$. For any $P \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$, define $\mathfrak{D}(P) := \mathfrak{D}(\langle P\rangle)$ and denote by $\deg(P)$ the maximum degree of elements of P.

Definition 4.1 ([35, Chap. 4, Def. 2.1]) Let $I \subseteq \mathbb{C}[\overline{\boldsymbol{x}}]$ be a zero-dimensional ideal and $p \in \mathtt{V}(I)$. The *multiplicity* of p, denoted $\mathcal{M}_I(p)$, is the dimension of the $\mathbb{C}$-vector space $\mathbb{C}[\overline{\boldsymbol{x}}]_p / I\mathbb{C}[\overline{\boldsymbol{x}}]_p$.

Theorem 4.2 ([35, Chap. 4, Cor. 2.5]) *Let $I \subseteq \mathbb{C}[\overline{\boldsymbol{x}}]$ be a zero-dimensional ideal. We have $\mathfrak{D}(I) = \sum_{\alpha \in \mathtt{V}(I)} \mathcal{M}_I(\alpha)$.*

Note that the most complicated computation in our algorithms is Gröbner bases computation of an ideal with its generator polynomials and of a saturated ideal. The arithmetic complexity of them is given in the following theorem and corollary, respectively.

Theorem 4.3 ([26, Prop. 8.1], [27, Thm. 3]) *Let $P \subseteq \mathbb{Q}[x_1, \cdots, x_n]$ be a zero-dimensional system, and G be the reduced Gröbner basis of $\langle P\rangle$ with respect to any monomial ordering. Then,*

(a) $\deg(G) \leq \mathfrak{D}(P)$, *and*

(b) *the arithmetic complexity of computing G, denoted $\mathtt{CGB}(n, \deg(P))$, can be polynomial in $\deg(P)^n$.*

Corollary 4.4 *Let $P \subseteq \mathbb{Q}[x_1, \cdots, x_n]$ be a zero-dimensional system, $f \in \mathbb{Q}[x_1, \cdots, x_n]$ and G be the reduced LEX Gröbner basis of $\langle P\rangle : f^\infty$. Then, the arithmetic complexity of computing G is $\mathtt{CGB}(n+1, \max(\deg(P), \deg(f)+1))$.*

Proof Let τ be a new variable and G_0 be the reduced LEX Gröbner basis of the ideal $\langle P \cup \{1-\tau f\}\rangle$ for $x_1 < \cdots < x_n < \tau$. By [34, Chap. 4.4, Thm. 14], we have $G = G_0 \cap \mathbb{Q}[x_1, \cdots, x_n]$. Therefore, the arithmetic complexity of computing G is equal to it of computing G_0, which is $\mathtt{CGB}(n+1, \max(\deg(P), \deg(f)+1))$ by Theorem 4.3. ∎

Remark that in Theorem 4.3, $\mathtt{CGB}(n, \deg(P))$ depends on algorithms used for computing Gröbner bases. And in Corollary 4.4, we only prove the complexity of computing the reduced LEX Gröbner basis of a saturated ideal, while that of computing the reduced Gröbner basis with respect to any monomial ordering can be used in our complexity analysis. Let P and f be the same as in Corollary 4.4. We notice that [31] shows an LEX Gröbner basis of $\langle P\rangle : f$ can be computed costing at most $O(\mathfrak{D}(P)^3)$ arithmetical operations. It seems that a reduced Gröbner basis of $\langle P\rangle : f^\infty$ may be computed within the same complexity. Naturally, if saturated ideals computation has better complexity, then the complexity of our algorithms will be better.

4.1 Complexity of Algorithm 2

Let $F \subseteq \mathbb{Q}[x_1, \cdots, x_n]$ be a zero-dimensional system, $d := \deg(F)$ and $D := \max(d, \mathfrak{D}(F))$ in the whole subsection. For convenience, we assume $\mathtt{V}(F) \neq \emptyset$.

Remark 4.5 By Bézout's theorem, $D \leq d^n$.

Theorem 4.6 *Algorithm 2 decomposes F into a finite set of pure chains with the arithmetic complexity*

$$(2\mathfrak{D}(F)-1) \cdot (\mathtt{CGB}(n, D) + \mathtt{CGB}(n, \mathfrak{D}(F))), \tag{1}$$

which can be polynomial in D^n.

In order to prove Theorem 4.6, we prepare some lemmas.

Given the input F, suppose that Algorithm 2 terminates with N ($N \geq 1$) times of loops. Let P_i be the picked polynomial set in the i-th loop. Consider a binary tree T with $P_1, \cdots, P_N$ as nodes. If $\langle P_i\rangle \neq \langle 1\rangle$ and the W-characteristic set of P_i is not pure, the node P_i has two child nodes $P_{i_1} = \{G \cup \{\mathtt{ini}(C_k)\}\}$ and $P_{i_2} = \{G \cup G_{sat}\}$ (see Line 16), where $i < i_j \leq N$ for $j = 1, 2$. Otherwise, the node P_i is a leaf. We denote by $\mathtt{child}(P_i)$ the set of all child nodes of P_i (if P_i is a leaf, $\mathtt{child}(P_i) = \emptyset$). The root node of tree T is $P_1 = F$.

For each node P_i of tree T, we define a value:

$$\mathtt{Value}(P_i) := \begin{cases} \mathfrak{D}(P_i), & \mathtt{V}(P_i) \neq \emptyset, \\ 1, & \mathtt{V}(P_i) = \emptyset. \end{cases} \tag{2}$$

Note that $\mathtt{V}(P_i) = \emptyset$ if and only if $\mathfrak{D}(P_i) = 0$, and thus $\mathfrak{D}(P_i) \leq \mathtt{Value}(P_i)$.

Lemma 4.7 *If $\langle F\rangle \subsetneq \langle G\rangle \subseteq \mathbb{C}[\overline{\boldsymbol{x}}]$, then*
(a) $\mathcal{M}_{\langle G\rangle}(p) \leq \mathcal{M}_{\langle F\rangle}(p)$ *for any $p \in \mathtt{V}(G)$, and*
(b) $\mathfrak{D}(G) < \mathfrak{D}(F)$.

Proof It is obvious by the definitions of multiplicity and dimension. ∎

Lemma 4.8 *For tree T, we have*

$$\sum_{P_{i_j} \in \mathtt{child}(P_i)} \mathtt{Value}(P_{i_j}) \leq \mathfrak{D}(P_i), \tag{3}$$

$$\sum_{P_i \textit{ is a leaf of } T} \mathtt{Value}(P_i) \leq \mathfrak{D}(F). \tag{4}$$

Proof We prove (3) first. If P_i is a leaf, it is clear. Otherwise, we have $\langle P_i\rangle \neq \langle 1\rangle$, i.e., $\mathtt{V}(P_i) \neq \emptyset$. Note that $\mathtt{V}(P_i) = \mathtt{V}(P_{i_1}) \cup \mathtt{V}(P_{i_2})$ by the proof of Theorem 3.7. Then, at least one variety of child nodes is not $\emptyset$. We also note that $\langle P_i\rangle \subsetneq \langle P_{i_j}\rangle$ for $j = 1, 2$ and $V(P_{i_1}) \cap V(P_{i_2}) = \emptyset$ by the proof of Theorem 3.7. Thus, if neither $\mathtt{V}(P_{i_1})$ nor $\mathtt{V}(P_{i_2})$ is an empty set, then by Theorem 4.2 and Lemma 4.7 (a),

$$\begin{aligned} \sum_{P_{i_j} \in \mathtt{child}(P_i)} \mathtt{Value}(P_{i_j}) &= \sum_{j=1,2} \mathfrak{D}(P_{i_j}) \\ &= \sum_{j=1,2} \sum_{p \in \mathtt{V}(P_{i_j})} \mathcal{M}_{\langle P_{i_j}\rangle}(p) \\ &\leq \sum_{p \in \mathtt{V}(P_i)} \mathcal{M}_{\langle P_i\rangle}(p) = \mathfrak{D}(P_i). \end{aligned} \tag{5}$$

If $\mathtt{V}(P_{i_1}) = \emptyset$ and $\mathtt{V}(P_{i_2}) \neq \emptyset$, then by Lemma 4.7 (b),

$$\sum_{P_{i_j} \in \mathtt{child}(P_i)} \mathtt{Value}(P_{i_j}) = 1 + \mathfrak{D}(P_{i_2}) \leq \mathfrak{D}(P_i). \tag{6}$$

By (5) and (6), (3) is proved. Then, it is clear that (4) holds by induction. ∎

Lemma 4.9 *For every node P_i $(i = 1, \cdots, N)$ of tree T, let G_i be the reduced LEX Gröbner basis of $\langle P_i\rangle$. Then,*
(a) $\deg(G_i) \leq \mathfrak{D}(F)$,
(b) $\deg(P_i) \leq \max(d, \mathfrak{D}(F))$.

Proof (a) By Theorem 4.3, $\deg(G_i) \leq \mathfrak{D}(P_i)$. Then, we prove $\mathfrak{D}(P_i) \leq \mathfrak{D}(F)$ by induction on i. It is clear for $i = 1$. Assume it holds for $1, \cdots, i-1$. By Lemma 4.8, $\mathtt{Value}(P_i) \leq \mathfrak{D}(P_{i^*})$,

where P_{i^*} is the parent of P_i. Since $i^* < i$, by the assumption, $\mathfrak{D}(P_{i^*}) \leq \mathfrak{D}(F)$. Thus, $\mathfrak{D}(P_i) \leq \mathtt{Value}(P_i) \leq \mathfrak{D}(P_{i^*}) \leq \mathfrak{D}(F)$.

(b) The maximum degree of elements of $P_1 = F$ is d. Note that each node P_j ($2 \leq j \leq N$) has a parent P_{j^*} ($1 \leq j^* < j$). So, P_j is one of $G \cup \{\mathtt{ini}(C_k)\}$ and $G \cup G_{sat}$ in Line 16, where G is the reduced LEX Gröbner basis of $\langle P_{j^*}\rangle$. We only need to prove that $\deg(G \cup \{\mathtt{ini}(C_k)\}) \leq \mathfrak{D}(F)$ and $\deg(G \cup G_{sat}) \leq \mathfrak{D}(F)$. By (a), we have $\deg(G) \leq \mathfrak{D}(F)$. Then, since $C_k \in G$, we have $\deg(\mathtt{ini}(C_k)) \leq \mathfrak{D}(F)$. It remains to prove $\deg(G_{sat}) \leq \mathfrak{D}(F)$. By Theorem 4.3 (a), $\deg(G_{sat}) \leq \mathfrak{D}(G_{sat})$. Because $\langle C_1, \cdots, C_{k-1}\rangle \subseteq \langle G_{sat}\rangle$, by Lemma 4.7 (b), $\mathfrak{D}(G_{sat}) \leq \mathfrak{D}(C_1, \cdots, C_{k-1})$. By Lemma 3.5, $\langle C_1, \cdots, C_{k-1}\rangle = \langle G \cap \mathbb{C}[x_1, \cdots, x_{k-1}]\rangle$. So, $\mathfrak{D}(\langle C_1, \cdots, C_{k-1}\rangle)$ is equal to the dimension of $\mathbb{C}[x_1, \cdots, x_{k-1}]/(\langle G\rangle \cap \mathbb{C}[x_1, \cdots, x_{k-1}])$. Note that $\mathbb{C}[x_1, \cdots, x_{k-1}]/(\langle G\rangle \cap \mathbb{C}[x_1, \cdots, x_{k-1}])$ is equal to the quotient ring $\mathbb{C}[x_1, \cdots, x_n]/\langle G\rangle$ limited on $\mathbb{C}[x_1, \cdots, x_{k-1}]$. Then, the dimension is at most $\mathfrak{D}(G)$, i.e., $\mathfrak{D}(P_{j^*})$. Note that $\mathfrak{D}(P_{j^*}) \leq \mathfrak{D}(F)$ by the proof of (a). We complete the proof. ∎

Lemma 4.10 *The number of leaves of tree T is at most $\mathfrak{D}(F)$. The number of nodes of tree T is at most $2\mathfrak{D}(F) - 1$, i.e., $N \leq 2\mathfrak{D}(F) - 1$.*

Proof Note that $\mathtt{Value}(P_i) \geq 1$ for $i = 1, \cdots, N$. Then, by (4) of Lemma 4.8, the number of leaves is at most $\mathfrak{D}(F)$. Note that every node is either a leaf or has two child nodes. So, the number of nodes is at most $2\mathfrak{D}(F) - 1$. ∎

Proof of Theorem 4.6 By Lemma 4.10, the number of times of loops $N \leq 2\mathfrak{D}(F) - 1$. In each loop, the most complicated computation is Gröbner bases computation in Line 5 and Line 15. By Lemma 4.9 (b), the computation in Line 5 has the complexity $\mathtt{CGB}(n, D)$. By Corollary 4.4 and Lemma 4.9 (a), the complexity of the computation in Line 15 is $\mathtt{CGB}(k, \mathfrak{D}(F))$. Note that $k \leq n$. Thus, (1) is proved. By Theorem 4.3 (b), (1) can be polynomial in D^n. ∎

4.2 Complexity of Algorithm 3 & Algorithm 1

Let F, d and D be the same as in Section 4.1. We also assume $\mathtt{V}(F) \neq \emptyset$.

Theorem 4.11 *An SFPTD of a reduced pure chain $\mathcal{T} \subseteq \mathbb{Q}[\overline{\boldsymbol{x}}]$ can be computed by Algorithm* 3 *within a complexity of polynomial in $\mathfrak{D}(\mathcal{T})^n$. An SFPTD of F can be computed by Algorithm* 1 *within a complexity of polynomial in d^{n^2}.*

To prove Theorem 4.11, we prepare some lemmas first.

Given a reduced pure chain $\mathcal{T}$, suppose that Algorithm 3 terminates with M ($M \geq 1$) times of loops. Let $\mathcal{P}_i$ be the picked reduced pure chain in the i-th loop (see Corollary 3.8). Consider a tree $\widetilde{T}$ with some nodes $\mathcal{P}_1, \cdots, \mathcal{P}_M$ and some other nodes $\{1\}$. If $\mathcal{P}_i$ is not square-free, then the node $\mathcal{P}_i$ has one or more reduced pure chains in Line 11 or Line 14 as child nodes. Otherwise, the node $\mathcal{P}_i$ has no child node. If $\mathcal{P}_i$ has and only has one child node, let the set $\{1\}$ be its second child node. The root node of tree $\widetilde{T}$ is $\mathcal{P}_1 = \mathcal{T}$. We also define a value of every node $\mathcal{P}_i$ as in (2).

Lemma 4.12 *For every node $\mathcal{P}$ of tree $\widetilde{T}$,* $\deg(\mathcal{P}) \leq \mathfrak{D}(\mathcal{P})$.

Proof If $\mathcal{P} = \{1\}$, the conclusion is clear. Otherwise, $\mathcal{P}$ is a reduced pure chain. Then, by

Proposition 3.2 and Theorem 4.3 (a), we complete the proof. ∎

Lemma 4.13 *For tree $\widetilde{T}$, we have $\sum_{\mathcal{P}_{i_j}\in \mathtt{child}(\mathcal{P}_i)} \mathtt{Value}(\mathcal{P}_{i_j}) \leq \mathfrak{D}(\mathcal{P}_i)$ and $\sum_{\mathcal{P}_i \text{is a leaf of } \tilde{T}} \mathtt{Value}(\mathcal{P}_i) \leq \mathfrak{D}(\mathcal{T})$.*

Proof Note that in Line 11, $\langle P_1, \cdots, P_n\rangle \subsetneq \langle \xi_i, P_2, \cdots, P_n\rangle$ for $i = 1, \cdots, m$, $\mathtt{V}(P_1, \cdots, P_n) = \bigcup_{i=1}^{m} \mathtt{V}(\xi_i, P_2, \cdots, P_n)$ and $\mathtt{V}(\xi_i, P_2, \cdots, P_n) \cap \mathtt{V}(\xi_j, P_2, \cdots, P_n) = \emptyset$ for $i \neq j$. Then, it is similar to the proof of Lemma 4.8. ∎

Lemma 4.14 *The number of nodes of tree $\widetilde{T}$ is at most $2\mathfrak{D}(\mathcal{T}) - 1$ which implies $M \leq 2\mathfrak{D}(\mathcal{T}) - 1$.*

Proof Note that if a node is not a leaf, then it has at least two child nodes. Then, it is similar to the proof of Lemma 4.10. ∎

Proof of Theorem 4.11 Firstly, we analyze the complexity of Algorithm 3. By Lemma 4.14, the number of loop steps $M \leq 2\mathfrak{D}(\mathcal{T})-1$. In each loop, the most complicated computation is in Line 13 and Line 14. By Corollary 4.4 and by Lemma 4.12, the complexity of computing G_{sat} in Line 13 is $\mathtt{CGB}(n+1, \mathfrak{D}(\mathcal{P}))$, where $\mathcal{P}$ is the reduced pure chain chose in Line 3. By Theorem 4.3 (b), it can be polynomial in $\mathfrak{D}(\mathcal{P})^n$. It remains to analyze the complexity of computation in Line 14. Similar to the proof of Lemma 4.9, we have $\mathfrak{D}(\mathcal{P}) \leq \mathfrak{D}(\mathcal{T})$ and $\deg(G_{sat}) \leq \mathfrak{D}(\mathcal{P})$. And by Lemma 4.12, $\deg(\mathcal{P}) \leq \mathfrak{D}(\mathcal{P})$. Thus, it is clear that $\deg(\mathcal{P} \cup \{\mathtt{sep}(P_k)\})$, $\mathfrak{D}(\mathcal{P} \cup \{\mathtt{sep}(P_k)\})$, $\deg(\mathcal{P} \cup G_{sat})$ and $\mathfrak{D}(\mathcal{P} \cup G_{sat})$ are all less than or equal to $\mathfrak{D}(\mathcal{T})$. Then, by Theorem 4.6, the complexity can be polynomial in $\mathfrak{D}(\mathcal{T})^n$. Therefore, after multiplying M, the complexity of Algorithm 1 can still be polynomial in $\mathfrak{D}(\mathcal{T})^n$.

Secondly, we analyze the complexity of Algorithm 1. By Theorem 4.6, the complexity of the calculation in Line 1 can be polynomial in D^n. Suppose the reduced pure chains (see Corollary 3.8) computed in Line 1 are $\mathcal{T}_1, \cdots, \mathcal{T}_t$. By Lemma 4.10, $t \leq \mathfrak{D}(F)$. By (4) of Lemma 4.8, $\mathfrak{D}(\mathcal{T}_i) \leq \mathtt{Value}(\mathcal{T}_i) \leq \mathfrak{D}(F)$. Thus, by the conclusion in the above paragraph, the complexity of the calculation in Line 5 can be polynomial in $\mathfrak{D}(F)^n$. Then, the complexity of Algorithm 1 can be polynomial in $\max(D^n, \mathfrak{D}(F)^n) = D^n$. Since $D < d^n$ (see Remark 4.5), the complexity can be polynomial in d^{n^2}. ∎

Recall that Theorem 4.6 and Theorem 4.11 talk about arithmetic complexity without analyzing the growth of the size of coefficients. In fact, the size of the coefficients in the algorithms may increase very fast.

5 Two Applications of SFPTD

In the section, we present two applications of SFPTD: Real solution isolation and computing radicals. Given a zero-dimensional system $F \subseteq \mathbb{Q}[\boldsymbol{x}]$, we first compute an SFPTD $\{\mathcal{T}_1, \cdots, \mathcal{T}_s\}$ of F by Algorithm 1.

In order to compute the isolating cubes of real solutions of F, we compute the isolating cubes of every square-free pure chain $\mathcal{T}_i$ by [7, Algorithm NREALZERO]. The method is called NRSI in Section 6.

We denote by $\sqrt{I}$ the radical of an ideal $I \subseteq \mathbb{C}[\boldsymbol{x}]$. We claim that $\sqrt{\langle F\rangle} = \bigcap_{i=1}^{s} \langle \mathcal{T}_i\rangle$.

The proof of the claim is as follows. Since $\{\mathcal{T}_1, \cdots, \mathcal{T}_s\}$ is an SFPTD of F, we have $\mathtt{V}(F) = \bigcup_{i=1}^{s} \mathtt{V}(\mathcal{T}_i)$. Then, $\mathtt{V}(F) = \mathtt{V}(\bigcap_{i=1}^{s} \langle \mathcal{T}_i \rangle)$. So, $\sqrt{\langle F \rangle} = \bigcap_{i=1}^{s} \sqrt{\langle \mathcal{T}_i \rangle}$. Note that every $\mathcal{T}_i$ is a square-free pure chain. Then, by [36, Corollary 3.3] and by Proposition 3.2 (a), $\langle \mathcal{T}_i \rangle$ is radical. Thus, we complete the proof. The method for computing $\sqrt{\langle F \rangle}$ by the intersection of ideals is called IRA in Section 6.

6 Experiments

We implemented Algorithm 1, the methods NRSI and IRA with `Maple2021`, where we use the `Maple` command `Groebner[Basis]` for computing Gröbner bases in Algorithm 2–Line 5&Line 15 and Algorithm 3–Line 13.

In the section, we explain implementation details and show the experimental results of partial testing examples. All testing examples, code and experimental results are available online via: `https://github.com/lihaokun/StrongSfTriDec`. All tests were conducted on 16-Core Intel Core i7-12900KF@3.20GHz with 128GB of memory and Windows 11.

6.1 Description of the Experimentation

Testing examples are collected from the literatures [7, 9, 18] and the website `http://homepages.math.uic.edu/~jan/demo.html`. We just get rid of the ones that are repeated or not zero-dimensional. Owing to space constraints, we only present 44 "difficult" examples (total 151 examples) in Table 1. Timings are in seconds. "OT" means out of the timing 3600 seconds, and "LOSS" means kernel connection lost during calculation of `Maple`. The column "sys" denotes the name of the polynomial system. The column "n/d" stands for the number of variables/the maximum degree of elements in the system.

We record the time to compute triangular decomposition by Algorithm 1 (see the column Algorithm 1) and two `Maple` commands (see the column mp-rc) in the group of columns TD. The two commands used are `RegularChains[Triangularize]` with the option `radical=yes`, and `RegularChains[ChainTools][SeparateSolutions]`. The first command, which is also used in [9], decomposes the system to a finite number of square-free regular chains. Although the zero sets of these square-free regular chains are pairwise disjoint in general, we ensure it again by the second command. In the experiment, the second command takes almost no time. In fact, since a pure chain is a regular chain, SFPTD is stronger than such decomposition.

In the group of columns RSI, we record the time of real solution isolation computed by the method NRSI (see the column NRSI), the `Mathematica12` command `Solve` (see the column mt-solve) and the `Maple` command `RootFinding[Isolate]` (see the column mp-rf).

In the group of columns RA, we record the time to compute radicals by the method IRA (see the column IRA) and the `Maple` command `PolynomialIdeals[Radical]` (see the column mp-radical).

6.2 Statistical Experimental Results

We show the statistical experimental results of all 151 examples in Table 2. The number of examples that can be solved within 3600 seconds is recorded in the row Solved. The number of

LOSS (OT) examples is recorded in the row LOSS (OT). We record the sum of the computing time of all solved examples (written as solved time) in the row Time (Solved). For every LOSS or OT example, we record their computing time as 3600 seconds. And, the sum of the computing time of all examples is recorded in the row Time.

For triangular decomposition, Algorithm 1 performs significantly better than mp-rc. There are 31 examples which can only be solved by Algorithm 1. And the solved time of Algorithm 1 is a half of that of mp-rc. One main reason why Algorithm 1 performs better is that the outputs of Algorithm 1 usually have less components. Denote by m_1 and m_2 the numbers of components computed by Algorithm 1 and mp-rc on the same example, respectively. We observe that

Table 1 Timings for computing triangular decomposition of zero-dimensional systems, isolating cubes of real solutions, and radicals of zero-dimesional ideals

		TD		RSI			RA	
sys	n/d	Algorithm 1	mp-rc	NRSI	mt-solve	mp-rf	IRA	mp-radical
nld-4-5	5/4	35.30	OT	35.64	1095.42	1522.17	368.91	88.79
nld-6-4	4/6	83.83	OT	86.06	103.09	2918.52	183.19	878.51
nld-9-3	3/9	22.22	0.65	47.50	2.71	191.98	39.02	OT
nld-10-3	3/10	109.90	0.50	130.42	12.39	670.93	117.18	OT
nql-10-4	10/4	0.01	0.09	0.14	0.00	OT	0.02	OT
nql-15-2	15/2	0.01	0.17	0.18	0.02	OT	0.01	OT
Reif	16/2	0.02	0.80	0.02	OT	0.07	0.02	0.17
simple-nql-20-30	20/30	0.01	0.28	0.66	0.12	OT	0.01	LOSS
Trinks-2	6/3	0.01	OT	0.02	0.01	0.11	0.03	0.04
Trinks-difficult	6/3	0.12	OT	0.16	0.02	0.31	0.18	0.26
Uteshev-Bikker	4/3	0.29	OT	0.63	0.25	1.16	0.49	1.22
wang_ex34	14/5	0.09	34.64	0.11	OT	0.21	0.14	0.07
wang_ex40	6/2	0.15	OT	0.77	0.37	0.36	0.26	0.65
boon	6/4	0.09	OT	0.19	0.02	0.13	0.13	0.35
cpdm5	5/3	19.25	OT	19.64	13.46	2.90	55.57	16.28
eco8	8/3	0.33	LOSS	0.75	2.03	0.27	0.44	1.49
redcyc6	6/11	1.01	OT	1.34	0.38	1.91	3.38	9.43
redcyc7	7/13	44.22	LOSS	55.25	OT	1684.53	570.98	166.93
extcyc6	6/6	5.88	OT	7.36	43.90	3.25	38.25	15.62
cassou	4/8	0.23	OT	0.29	0.06	0.36	0.35	0.76
virasoro	8/2	125.48	OT	128.64	OT	31.66	2477.53	146.24
d1	12/3	4.90	OT	8.37	0.77	6.03	13.95	675.68
kin1	12/3	22.00	OT	39.90	1.16	19.69	39.74	902.41
des18_3	8/3	22.15	OT	22.83	1.74	1.99	32.89	3.61
kinema	9/2	0.85	OT	1.27	1.34	0.56	1.19	2.78
rbpl24	9/2	14.54	OT	19.82	6.54	10.87	20.33	408.90

Table 1 (Continued) Timings for computing triangular decomposition of zero-dimensional systems, isolating cubes of real solutions, and radicals of zero-dimesional ideals

		TD		RSI			RA	
sys	n/d	Algorithm 1	mp-rc	NRSI	mt-solve	mp-rf	IRA	mp-radical
reimer5	5/6	0.42	OT	0.97	5.17	1.55	0.60	5.29
filter9	9/4	2.66	OT	6.33	4.73	36.24	3.85	45.37
katsura6	7/2	0.99	OT	2.92	2.37	0.47	1.35	8.46
katsura7	8/2	12.42	LOSS	33.09	40.46	1.68	15.18	29.53
katsura8	9/2	291.76	OT	757.49	920.58	13.24	340.98	OT
katsura9	10/2	OT	LOSS	OT	OT	151.71	OT	OT
katsura10	11/2	OT	OT	OT	OT	2123.81	OT	OT
utbikker	4/3	1.25	OT	1.43	0.24	1.31	2.13	3.35
kotsireas	6/5	4.16	OT	4.41	2.81	1.59	8.77	6.91
chandra6	6/2	1.94	OT	3.27	0.37	0.75	3.03	1.82
tangents0	6/2	0.92	OT	1.07	0.10	0.69	1.38	0.76
assur44	8/3	4.79	OT	5.95	10.65	2.79	6.33	7.11
cyclic6	6/6	1.18	OT	1.52	1.02	1.24	3.93	12.68
cyclic7	7/7	76.70	OT	90.17	OT	691.37	859.50	290.58
cyclic9	9/9	OT	OT	OT	OT	OT	OT	OT
cyclic10	10/10	OT	LOSS	OT	OT	OT	OT	OT
cyclic11	11/11	OT	LOSS	OT	OT	OT	OT	OT
kss3	10/2	212.92	212.86	216.60	2.67	249.54	OT	540.37

Table 2 Statistical experimental results of all testing examples (151 examples)

	TD		RSI			RA	
	Algorithm 1	mp-rc	NRSI	mt-solve	mp-rf	IRA	mp-radical
Solved	146	115	146	141	145	145	140
LOSS	0	6	0	0	0	0	0
OT	5	30	5	10	6	6	11
Time (Solved)	1191.37	2842.90	1842.40	2301.10	11238.8	5438.7	6695.10
Time	19191.37	110842.90	19842.40	38301.10	32838.8	27038.7	42695.10

$m_1 < m_2$ for 61 examples. Especially, for 54 of those 61 examples, we have $m_1 \leq \frac{1}{2}m_2$. On the contrary, there are no examples where $m_1 > m_2$. And $m_1 = m_2$ for 54 examples where $m_1 = m_2 = 1$ for 39 examples.

For real solution isolation, mt-solve performs better than NRSI on small examples which can be solved in 2 seconds, but the solved time of NRSI is approximately 450 seconds less than that of mt-solve. And, there are 5 difficult examples solved successfully by NRSI which cannot be solved by mt-solve. NRSI solves 3 examples that mp-rf does not, while mp-rf solves

2 examples that NRSI does not. However, the solved time of NRSI is approximately 10000 seconds less than that of mp-rf. It is worth noting that for the system katsura8, the computing time of NRSI is three times that of SFPTD. This is because the computed SFPTD has huge coefficients.

To compute radicals, the method IRA solves 145 examples successfully, while mp-radical solves 140. The solved time of IRA is about 1200 seconds less than that of mp-radical. Since it is difficult to compute intersections of ideals (see the systems redcyc7 and kss3), IRA does not perform as well as we expect.

7 Conclusion

In the paper, we propose an algorithm for computing SFPTD and prove that the arithmetic complexity can be single exponential time (note that there are few results about the complexity of triangular-decomposition algorithms). Our algorithm is partly inspired by [19, Algorithm 1] and thus it is based on Gröbner bases. The novelty of our algorithm is that we make use of separant and saturated ideals to ensure that every pure chain is square-free and the zero sets of any two pure chains have no intersection, respectively. It is worth noting that although SFPTD is stronger than zs-rc decomposition in [9], our algorithm is much more efficient than the classical method in experiments. The only disadvantage of our algorithm is that a computed SFPTD of a big system always has huge coefficients. So, it sometimes takes a large amount of time to compute isolating cubes of every square-free pure chain or compute intersections of ideals. We will consider giving a bit complexity analysis of our algorithm in the future.

Conflict of Interest

The authors declare no conflict of interest.

References

[1] Wu W, On the decision problem and the mechanization of theorem-proving in elementary geometry, *Scientia Sinica*, 1978, **21**(2): 159–172.

[2] Wu W, Basic principles of mechanical theorem proving in elementary geometries, *Journal of Systems Science and Mathematical Sciences*, 1984, **4**(3): 207–235.

[3] Chou S and Gao X, Ritt-wu's decomposition algorithm and geometry theorem proving, *CADE* 1990, Springer, 1990, 207–220.

[4] Yang L and Zhang J, Searching dependency between algebraic equations: An algorithm applied to automated reasoning, Technical report, ICTP, 1991.

[5] Yang L, Zhang J, and Hou X, A criterion of dependency between algebraic equations and its applications, *Proc. IWMM*, 1992, 110–134.

[6] Xia B and Yang L, An algorithm for isolating the real solutions of semi-algebraic systems, *J. Symb. Comput.*, 2002, **34**(5): 461–477.

[7] Xia B and Zhang T, Real solution isolation using interval arithmetic, *Comput. Math. Appl.*, 2006, **52**(6–7): 853–860.

[8] Wu W and Gao X, Automated reasoning and equation solving with the characteristic set method, *J. Comput. Sci. Tech.*, 2006, **21**(5): 756–764.

[9] Boulier F, Chen C, Lemaire F, et al., Real root isolation of regular chains, *Computer Mathematics*, Springer Berlin Heidelberg, 2014: 33–48.

[10] Gao X, An introduction to wu's method of mechanical geometry theorem proving, *Proc. IFIP TC*12/*WG*12.3, 1992, 13–22.

[11] Cheng J, Gao X, and Guo L, Root isolation of zero-dimensional polynomial systems with linear univariate representation, *J. Symb. Comput.*, 2012, **47**(7): 843–858.

[12] Xia B and Yang L, *Automated Inequality Proving and Discovering*, World Scientific, Singapore, 2016.

[13] Chen Z, Tang X, and Xia B, Generic regular decompositions for parametric polynomial systems, *Journal of Systems Science & Complexity*, 2015, **28**(5): 1194–1211.

[14] Wang D, *Elimination Methods*, Springer Science & Business Media, 2001.

[15] Aubry P, Lazard D, and Maza M M, On the theories of triangular sets, *J. Symb. Comput.*, 1999, **28**(1–2): 105–124.

[16] Kalkbrener M, A generalized euclidean algorithm for computing triangular representations of algebraic varieties, *J. Symb. Comput.*, 1993, **15**(2): 143–167.

[17] Wang D, Zero decomposition algorithms for systems of polynomial equations, *Computer Mathematics*, World Scientific, Singapore, 2000, 67–70.

[18] Wang D, Solving polynomial equations: Characteristic sets and triangular systems, *Math. Comput. Simulation*, 1996, **42**(4–6): 339–351.

[19] Dong R and Mou C, On characteristic decomposition and quasi-characteristic decomposition, *CASC* 2019, Cham: Springer International Publishing, 2019, 122–139.

[20] Wang D, Computing triangular systems and regular systems, *J. Symb. Comput.*, 2000, **30**(2): 221–236.

[21] Dong R and Mou C, Decomposing polynomial sets simultaneously into gröbner bases and normal triangular sets, *CASC* 2017, Springer International Publishing, 2017: 77–92.

[22] Chen C, Solving polynomial systems via triangular decomposition, PhD thesis, The University of Western Ontario, London, 2011.

[23] Hubert E, Notes on triangular sets and triangulation-decomposition algorithms i: Polynomial systems, *International Conference on Symbolic and Numerical Scientific Computation*, Springer, 2001, 1–39.

[24] Chen C and Maza M M, Algorithms for computing triangular decomposition of polynomial systems, *J. Symb. Comput.*, 2012, **47**(6): 610–642.

[25] Buchberger B, Ein algorithmus zum auffinden der basiselemente des restklassenringes nach einem nulldimensionalen polynomideal, PhD thesis, Universitat Insbruck, 1965.

[26] Hashemi A and Lazard D, Complexity of zero-dimensional gröbner bases, PhD thesis, INRIA, 2005.

[27] Lakshman Y N, A single exponential bound on the complexity of computing gröbner bases of zero dimensional ideals, *Effective Methods in Algebraic Geometry*, Springer, Birkhäuser Boston, 1991, 227–234.

[28] Faugere J C, A new efficient algorithm for computing gröbner bases (f4), *J. Pure Appl. Algebra*,

1999, **139**(1–3): 61–88.

[29] Faugere J C, A new efficient algorithm for computing gröbner bases without reduction to zero (f5), *Proc. ISSAC'02*, 2002, 75–83.

[30] Lazard D, Solving zero-dimensional algebraic systems, *J. Symb. Comput.*, 1992, **13**(2): 117–131.

[31] Möller H M, On decomposing systems of polynomial equations with finitely many solutions, *Applicable Algebra in Engineering, Communication and Computing*, 1993, **4**(4): 217–230.

[32] Wang D, On the connection between ritt characteristic sets and buchberger-gröbner bases, *Math. Comput. Sci.*, 2016, **10**(4): 479–492.

[33] Becker T and Weispfenning V, *Gröbner Bases*, Springer, New York, 1998.

[34] Cox D, Little J, and OShea D, *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*, 4th ed, Springer Science & Business Media, Berlin, 2013.

[35] Cox D A, Little J B, and OShea D, *Using Algebraic Geometry*, Graduate texts in mathematics, 2nd Edition, Springer Science & Business Media, 2005.

[36] Boulier F, Lemaire F, and Maza M M, Well known theorems on triangular systems and the d5 principle, *TC*, 2006, 79–91.