Choosing the Variable Ordering for Cylindrical Algebraic Decomposition via Exploiting Chordal Structure

Haokun Li, Bican Xia, Huiying Zhang, and Tao Zheng
haokunli,huiyingz,1601110051@pku.edu.cn
xbc@math.pku.edu.cn
School of Mathematical Sciences, Peking University
Beijing, China

ABSTRACT

Cylindrical algebraic decomposition (CAD) plays an important role in the field of real algebraic geometry and many other areas. As is well-known, the choice of variable ordering while computing CAD has a great effect on the time and memory use of the computation as well as the number of sample points computed. In this paper, we indicate that typical CAD algorithms, if executed with respect to a special kind of variable orderings (called "the perfect elimination orderings"), naturally preserve chordality, which is an important property on sparsity of variables. Experimentation suggests that if the associated graph of the polynomial system in question is chordal (*resp.*, is nearly chordal), then a perfect elimination ordering of the associated graph (*resp.*, of a minimal chordal completion of the associated graph) can be a good variable ordering for the CAD computation. That is, by using the perfect elimination orderings, the CAD computation may produce a much smaller full set of projection polynomials than by using other naive variable orderings. More importantly, for the complexity analysis of the CAD computation via a perfect elimination ordering, an (m, d)-property of the full set of projection polynomials obtained via such an ordering is given, through which the "size" of this set is characterized. This property indicates that when the corresponding perfect elimination tree has a lower height, the full set of projection polynomials also tends to have a smaller "size". This is well consistent with the experimental results, hence the perfect elimination orderings with lower elimination tree height are further recommended to be used in the CAD projection.

KEYWORDS

Cylindrical algebraic decomposition, chordality, variable ordering, polynomial.

ACM Reference Format:
Haokun Li, Bican Xia, Huiying Zhang, and Tao Zheng. 2021. Choosing the Variable Ordering for Cylindrical Algebraic Decomposition via Exploiting Chordal Structure. In *International Symposium on Symbolic and Algebraic Computation (ISSAC '21), July 18–22, 2021, Saint Petersburg, .* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/...

ISSAC '21, July 18–22, 2021, Saint Petersburg, Russia

ACM ISBN 978-1-4503-6084-5/19/07…$15.00
https://doi.org/10.1145/...

1 INTRODUCTION

Cylindrical algebraic decomposition (CAD) has been widely used in real algebraic geometry and beyond since it was introduced by Collins in 1975 [13]. A CAD of an Euclidean space $\mathbb{R}^n$ for a given polynomial set $P \subset \mathbb{Z}[x_1, \ldots, x_n]$ is a cylindrical partition of the space $\mathbb{R}^n$ so that every polynomial in the set P is sign-invariant in each component (called "cell") of this partition. There are many CAD algorithms and their variants, see for example [4, 5, 7, 9, 13, 14, 18–22, 25, 26, 31]. A typical CAD algorithm usually contains two parts: the *projection* part and the *lifting* part. In the projection part, one eliminates the variables of the polynomial set P successively in some order. Then the sample points obtained in some one-dimensional space via the full set of projection polynomials are lifted step-by-step to sample points in the original (n-dimensional) space during the lifting part and each sample point in $\mathbb{R}^n$ represents a cell of the CAD.

It is long known that the variable ordering used in the projection part has a great effect on the time and memory use of the CAD computation as well as the number of sample points computed. There are many researches focusing on the problem of choosing a good variable ordering for the CAD computation. For instance, in [3, 6, 16], various heuristics are provided to suggest a relatively good variable ordering based on different rules. Latter in [10, 17, 23], the methods based on machine learning and artificial neural network are used to choose a good variable ordering for CAD computation.

The present paper is inspired by the previous work [8, 12, 27, 28] on combining the *chordal structure* with triangular decomposition of polynomial sets. In [12], based on the computation of triangular decomposition, Cifuentes and Parrilo compute the chordal network of a polynomial set by exploiting the chordal structure of its associated graph. Latter in [27], Mou *et al* indicate that Wang's algorithm and a subresultant-based algorithm, both in top-down style, for computing the triangular decomposition of a polynomial set preserve the chordal structure. And another subresultant-based algorithm for computing the regular decomposition in the same style is also proved to preserve the chordal structure. Then in [28] by Mou and Lai, it is proved that Wang's algorithm for computing the simple decomposition of a polynomial set in top-down style preserves the chordal structure of the polynomial set as well. Recently in [8], Chen proves that an incremental algorithm for computing the triangular decomposition of a polynomial set preserves the chordal structure, too. On the contrary, Cifuentes and Parrilo indicate that the computation of Gröbner basis of a polynomial set seems to violate the chordal structure [11].

In this paper we take advantage of the chordal structure of the associated graph of a polynomial set while computing CAD for this set to exploit its variable sparsity. We first indicate that all the basic operations usually used in the CAD projection algorithms, such as resultant, discriminant, subresultant, *etc*, preserve chordal structures. Based on this, it is an easy corollary that typical CAD algorithms naturally preserve chordal structures (if executed with respect to a perfect elimination ordering), thus also the variable sparsity pattern embedded in the chordality. Some experimental results show that a perfect elimination ordering of a chordal structure of the polynomial set in question may be a good variable ordering for computing CAD compared with other naive ones. More importantly, for the complexity analysis of the CAD computation via a perfect elimination ordering, an (m, d)-property for the full set of projection polynomials obtained by using such an ordering is provided, through which the "size" of this set is characterized. According to that property, when the corresponding perfect elimination tree has a lower height, the full set of projection polynomials tends to have a smaller "size". This is further supported by some experimental results. And because of that, the perfect elimination orderings with lower elimination tree height are further recommended while computing the CAD projection.

The rest of this paper is organized as follows: Section 2 includes some basic definitions. In Sections 3 and 4, we prove that typical CAD algorithms preserve the chordal structure of the associated graph of the polynomial set in question. In Section 5, the "size" of the full set of projection polynomials obtained in accordance with a perfect elimination ordering is characterized via an (m, d)-property, while Section 6 is devoted to showing the effectiveness of our approach by some examples from applications. Section 7 concludes the paper.

2 PRELIMINARIES

Denote by $\mathbb{K}[\bar{x}] = \mathbb{K}[x_1, \ldots, x_n]$ the polynomial ring in variables $x_1, \ldots, x_n$ with coefficients in a domain $\mathbb{K}$. For any x_k, a nonzero polynomial $f \in \mathbb{K}[\bar{x}]$ can be written in the form $f = \sum_{i=0}^{s} a_i x_k^i$ with $a_s \not\equiv 0, a_{s-1}, \ldots, a_0 \in \mathbb{K}[x_1, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n]$ and $s \in \mathbb{Z}_{\geq 0}$. The number s is called the *degree* of f with respect to (*w.r.t.*) the variable x_k and is denoted by $\mathtt{deg}(f, x_k)$, while those a_i's are called the *coefficients* of f *w.r.t.* x_k and they form a set $\mathtt{coeff}(f, x_k) = \{a_s, a_{s-1}, \ldots, a_0\}$. In particular, we call a_s the *leading coefficient* of f *w.r.t.* x_k and denote it by $\mathtt{lc}(f, x_k)$. The *content* of f *w.r.t.* x_k, denoted by $\mathtt{cont}(f, x_k)$, refers to the GCD of the coefficients $a_0, a_1, \ldots a_s$. The polynomial f is called *primitive w.r.t.* x_k if $\mathtt{cont}(f, x_k) = 1$ while the polynomial $f/\mathtt{cont}(f, x_k)$ is defined to be *the primitive part* of f *w.r.t.* the variable x_k. In addition, we denote by $\mathtt{cont}(F, x_k)$ the set of those non-constant contents of the elements of a polynomial set F.

In the following, we recall the definition of the associated graph of a polynomial set and the concept of chordal graph. Let f be a polynomial in $\mathbb{K}[\bar{x}]$ and denote by $\mathtt{var}(f)$ the set of variables which appear in f. For a polynomial set F, we define $\mathtt{var}(F) = \cup_{f \in F} \mathtt{var}(f)$. Then we have the following definition:

Definition 2.1. ([27], Def.1) Let $F \subset \mathbb{K}[\bar{x}]$ be a finite polynomial set. The undirected graph with the vertex set $\mathtt{var}(F)$ and the edge set

$$\{(x_i, x_j) \mid \exists f \in F, x_i, x_j \in \mathtt{var}(f)\}$$

is called *the associated graph* of F and is denoted by $\mathscr{G}(F)$.

Remark 1. *If* $\mathtt{var}(F) = \emptyset$, *then* $\mathscr{G}(F)$ *is the empty graph.*

Example 2.2. Set $F_1 = \{x_1x_3 - 1, x_1x_2 - 1, x_2x_3 - 1\}$ and $F_2 = \{y_1^4 - 1, y_1^2 + y_3, y_2^2 + y_3, y_3^2 + y_4\}$. Then the associated graphs $\mathscr{G}(F_1)$ and $\mathscr{G}(F_2)$ of them are shown in Fig. 2 and Fig. 3, respectively.

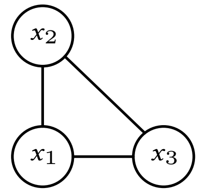

Figure 2: $\mathscr{G}(F_1)$

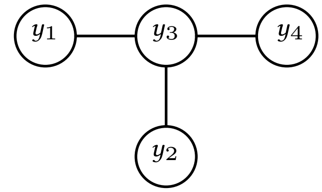

Figure 3: $\mathscr{G}(F_2)$

Definition 2.3. ([2]) Let $G = (V, E)$ be a graph with $V = \{x_1, \ldots, x_n\}$. An ordering of the vertices $x_{l_1} > x_{l_2} > \cdots > x_{l_n}$ is called *a perfect elimination ordering* if for each vertex x_l, the induced subgraph with the vertex set

$$X_l = \{x_l\} \cup \{x_k \mid x_k < x_l, \ (x_k, x_l) \in E\}$$

is a clique. A graph G with a perfect elimination ordering is said to be *chordal*.

The graphs $\mathscr{G}(F_1)$ and $\mathscr{G}(F_2)$ in Fig. 2 and Fig. 3 are both chordal graphs. In the graph $\mathscr{G}(F_1)$, any vertex ordering is a perfect elimination ordering. The graph $\mathscr{G}(F_2)$ is a tree, we know that trees are chordal with perfect elimination orderings obtained by randomly pruning the leafs.

Definition 2.4. ([1]) Let $G = (V, E)$ be a graph, *a chordal completion* of G is a chordal graph $\hat{G} = (\hat{V}, \hat{E})$ such that G is a subgraph of $\hat{G}$, *i.e.* $V \subset \hat{V}$ and $E \subset \hat{E}$. We call $\hat{G}$ a minimal chordal completion of G, if any subgraph of $\hat{G}$ is not a chordal completion of G.

For a given graph, there are some known algorithms, *e.g.* the *elimination game* algorithm in [29], to find one of its chordal completions and a corresponding perfect elimination ordering of that chordal completion.

Definition 2.5. Given a polynomial set $F \subset \mathbb{K}[\bar{x}]$, a graph G is *a chordal structure* of F if G is a chordal completion of $\mathscr{G}(F)$. In particular, if $\mathscr{G}(F)$ is a chordal structure of F, then F is said to be *chordal*. For convenience, any chordal graph is regarded as a chordal structure of $\mathscr{G}(F)$ if $\mathtt{var}(F) = \emptyset$.

3 BASIC OPERATIONS PRESERVING CHORDAL STRUCTURE

The propositions presented in this section show that some basic polynomial operations, such as discriminant, resultant and subresultant, preserve the chordal structure. Then it is reasonable that any algorithm, containing only these operations, also preserves the chordal structure. Now that the chordal structure is preserved by the algorithm, the variable sparsity embedded in the chordal structure is preserved as well. Later we will see the advantages of preserving such a sparsity pattern during the computation of CAD.

For any $f, g \in \mathbb{K}[\bar{y}, x] = \mathbb{K}[y_1, \ldots, y_m, x]$, we denote by $\mathtt{dis}(f, x)$ the discriminant of f *w.r.t.* x, and set $\mathtt{res}(f, g, x)$ to be the Sylvester resultant, S_j to be the j-th subresultant and $S_{\mu+1}, S_\mu, S_{\mu-1}, \cdots, S_0$

to be the subresultant chain of f and g *w.r.t.* x, respectively. Then we have

Proposition 3.1. *Let f be a polynomial in $\mathbb{K}[\bar{y}, x]$ and $G = (V, E)$ a chordal structure of the set $\{f\}$, then G is a chordal structure of the set $\mathfrak{S} = \texttt{coeff}(f, x) \cup \{\texttt{cont}(f, x), \texttt{dis}(f, x)\}$. In particular, G is a chordal structure of the set $\{\texttt{lc}(f, x)\}$.*

Proof. Since $\texttt{var}(\mathfrak{S}) \subset \texttt{var}(f)$, $\mathscr{G}(\mathfrak{S})$ is a subgraph of $\mathscr{G}(\{f\})$, which is a subgraph of G. □

Proposition 3.2. *Let f, g be polynomials in $\mathbb{K}[\bar{y}, x]$ and $G = (V, E)$ a chordal structure of the set $\{f, g\}$. If there is a perfect elimination ordering of G such that $x > y_i$ for any $y_i \in \bar{y}$, then G is a chordal structure of the set $\mathfrak{S} = \{\texttt{res}(f, g, x), S_{\mu+1}, S_\mu, S_{\mu-1}, \cdots, S_0\}$.*

Proof. Since $x > y_i$ for any $y_i \in \bar{y}$, $\texttt{var}(\mathfrak{S}) \subset \texttt{var}(\{f, g\}) \subset X_x$ with $X_x = \{x\} \cup \{z \in V \mid z < x, (x, z) \in E\}$ a clique of G. So G is a chordal completion of the graph $\mathscr{G}(\mathfrak{S})$. □

The *finest squarefree basis* of $A \subset \mathbb{K}[\bar{x}]$, denoted by $\mathscr{F}(A)$, is the set of all the irreducible factors of the elements of A (see [13], P.146). Then the following observation is clear:

Proposition 3.3. *If a finite set $A \subset \mathbb{K}[\bar{x}]$ has a chordal structure G, then G is also a chordal structure of the set $\mathscr{F}(A)$ and the set $A \cup \mathscr{F}(A)$.*

Proof. Clearly we have $\texttt{var}(\mathscr{F}(A)) \subset \texttt{var}(A)$. Since any polynomial in $\mathscr{F}(A)$ is either a polynomial in A or a factor of a polynomial in A, $\mathscr{G}(\mathscr{F}(A))$ is a subgraph of $\mathscr{G}(A)$, which is again a subgraph of G. □

4 CAD PROJECTIONS WITH PERFECT ELIMINATION ORDERINGS

In this section, we first prove that some well-known CAD algorithms, with the projection operators used in them preserving the chordal structure, also preserve the chordal structure themselves if the projection is done in accordance with a perfect elimination ordering. Based on this, the perfect elimination orderings are recommended while computing the CAD projection. Then, through two examples, we compare the size of the full sets of projection polynomials obtained by the CAD algorithms with respect to perfect elimination orderings and other random variable orderings. The examples show that the perfect elimination orderings can result in much smaller polynomial sets than other naive ones do.

A key concept in the CAD algorithms is the projection operator. In an abstract sense, the projection operator $\texttt{Proj}$ is a mapping which maps an n-dimensional polynomial set P containing some variable x to an $(n-1)$-dimensional polynomial set P' such that $x \notin \texttt{var}(P')$ and any P'-sign-invariant CAD of $\mathbb{R}^{n-1}$ can be induced by a P-sign-invariant CAD of $\mathbb{R}^n$. The *projection procedure* of P with variable ordering $x_n > \cdots > x_1$ is defined to be the sequence of polynomial sets $\{P_n, P_{n-1}, \ldots, P_1\}$, where $P_n = P$ and each $P_i = \texttt{Proj}(P_{i+1}, x_{i+1})$ is a set of polynomials in $x_i, \ldots, x_1$ for $i = n-1, \ldots, 1$. In particular, we call $P_{n-1}, \ldots, P_1$ the *projection polynomial sets w.r.t.* the ordering $x_n > \cdots > x_1$.

The projection operator introduced by Collins is defined to be a union of some coefficients, some resultants, some discriminants and some subresultants with respect to a fixed variable x. Many improved projection operators are obtained by simplifying Collins' projection operator.

Proposition 4.1. *Set $A \subset \mathbb{K}[\bar{y}, x]$ to be a polynomial set. Suppose that a projection operator $\texttt{Proj}(A, x)$ only consists of some coefficients, contents, resultants, discriminants and some subresultants of the polynomials in $A \cup \mathscr{F}(A)$ w.r.t. x and G is a chordal structure of A with a perfect elimination ordering such that $x > y_i$ for any $y_i \in \bar{y}$, then G is also a chordal structure of the polynomial set $\texttt{Proj}(A, x)$.*

Proof. Set $f \in \texttt{Proj}(A, x)$. If f is the discriminant, the content or a coefficient of a polynomial in $A \cup \mathscr{F}(A)$, then G is a chordal structure of $\{f\}$ by Prop. 3.3 and Prop. 3.1. If f is a resultant or a subresultant of two polynomials in $A \cup \mathscr{F}(A)$, then by Prop. 3.3 and Prop. 3.2, the same conclusion holds. Hence G is a chordal structure of any $\{f\} \subset \texttt{Proj}(A, x)$, thus also a chordal structure of the set $\texttt{Proj}(A, x)$. □

Proposition 4.2. *Set $A \subset \mathbb{K}[\bar{x}]$ to be a polynomial set. Suppose the graph G is a chordal structure of A and $x_n > \cdots > x_1$ is a perfect elimination ordering of G. If a projection operator $\texttt{Proj}(S, \cdot)$, with S the polynomial set on which it operates, consists of some coefficients, contents, resultants, discriminants and some subresultants of the polynomials in $S \cup \mathscr{F}(S)$ and $\{P_n = A, P_{n-1}, \ldots, P_1\}$ is a projection procedure of A obtained via the projection ordering $x_n > \cdots > x_1$, then G is a chordal structure of any P_i, $i = 1, \ldots, n$.*

Proof. The proof is inductive on the index i: If $i = n$, then $P_i = A$, the conclusion holds. Suppose that G is a chordal structure of P_k for some $k \le n$. Since $P_{k-1} = \texttt{Proj}(P_k, x_k)$, G is a chordal structure of P_{k-1} by Prop. 4.1. The proposition is proved. □

As can be seen from above, the original projection operator and most of the improved projection operators used in the CAD algorithms preserve the chordal structure of a polynomial set. For this reason, when the polynomial system is sparse and with the chordal structure, the perfect elimination orderings of that chordal structure are recommended while computing the CAD projection in order to preserve the chordal structure (as well as the sparsity embedded in the chordal structure) of the system. In this way, the growth of the size of the polynomial sets obtained during the projection can possibly be reduced.

In the following we take care of two popular projection operators and indicate that they preserve the chordal structure.

Definition 4.3. ([25]) Let $A \subset \mathbb{Z}[\bar{x}]$ be a finite polynomial set containing at least two variables. McCallum's projection operator $\texttt{Proj}_{\text{mc}}(\cdot, \cdot)$ is defined as follows: Let the polynomial set B be the finest squarefree basis of all the primitive parts of the elements of A with positive degrees, then

$$\texttt{Proj}_{\text{mc}}(A, x_n) = \texttt{cont}(A, x_n) \cup \bigcup_{f, g \in B, f \neq g} \{\texttt{res}(f, g, x_n)\} \cup \bigcup_{f \in B} \big((\texttt{coeff}(f, x_n) \backslash \{0\}) \cup \{\texttt{dis}(f, x_n)\}\big).$$

Since $B \subset \mathscr{F}(A)$, Prop. 4.2 holds for the operator $\texttt{Proj}_{\text{mc}}$. Indeed, by Prop. 3.3 and Prop. 4.1, each set in the projection procedure of any $A \subset \mathbb{Z}[\bar{x}]$ obtained by the operator $\mathscr{F}(\texttt{Proj}_{\text{mc}}(\cdot, \cdot))$ preserves the chordal structure of A as well.

Example 4.4. Consider the quantified formula below:

$$\exists x_1.\ x_1 + x_4 > 0 \wedge x_2 + x_4 \geq 0 \wedge x_3^2 + x_2 < 0$$
$$\wedge\, x_3^3 + x_1 \leq 0 \wedge x_5 + x_2 > 0 \wedge x_5 + x_1 + x_2 < 0.$$

The CAD-based methods solving this example rely on constructing a CAD for the set of polynomials

$$F = \{x_1 + x_4, x_2 + x_4, x_3^2 + x_2, x_3^3 + x_1, x_5 + x_2, x_5 + x_1 + x_2\}.$$

The associated graph of F is a chordal graph, shown in Fig. 3.

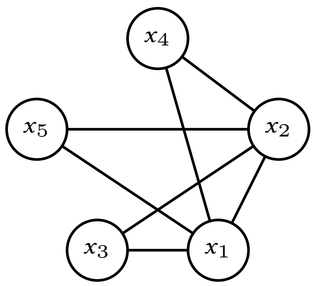

Figure 3: $\mathscr{G}(F)$

We successively use the operator $\mathscr{F}(\mathtt{Proj}_{\mathrm{mc}}(\cdot,\cdot))$ in accordance with the perfect elimination ordering $x_4 > x_5 > x_3 > x_2 > x_1$ and obtain a projection procedure:

$$\begin{aligned}
F' =& \mathscr{F}(\mathtt{Proj}_{\mathrm{mc}}(F, x_4))\\
=& \{x_1, x_2, x_3^2 + x_2, x_3^3 + x_1, x_5 + x_2, x_5 + x_1 + x_2, x_1 - x_2\},\\
F'' =& \mathscr{F}(\mathtt{Proj}_{\mathrm{mc}}(F', x_5))\\
=& \{x_3^2 + x_2, x_3^3 + x_1, x_1 - x_2, x_1 + x_2, x_1, x_2\},\\
F''' =& \mathscr{F}(\mathtt{Proj}_{\mathrm{mc}}(F'', x_3))\\
=& \{x_1 - x_2, x_1 + x_2, x_1, x_2, x_2^3 + x_1^2\},\\
F'''' =& \mathscr{F}(\mathtt{Proj}_{\mathrm{mc}}(F''', x_2))\\
=& \{x_1, x_1 + 1, x_1 - 1\}.
\end{aligned}$$

When a random variable ordering, say, $x_1 > x_2 > x_3 > x_4 > x_5$ is used instead, we finally obtain:

$$\begin{aligned}
F'''' = \{&x_5, x_5 - 1, x_5 + 1, x_5 - 2, x_5 - 8, 4x_5 - 1, 8x_5 - 1, 27x_5 - 4,\\
&3x_5 - 1, x_5^2 + x_5 - 1, x_5^2 + x_5 + 1, x_5^2 - x_5 + 1, x_5^2 - 3x_5 + 9,\\
&x_5^3 - x_5^2 + 2x_5 - 1, x_5^5 - 3x_5^4 + 3x_5^3 + 5x_5^2 + 2x_5 - 1,\\
&x_5^4 + x_5^3 + x_5^2 + x_5 + 1, x_5^5 + 4x_5^3 - x_5^2 + 2x_5 - 1,\\
&-x_5^3 + 4x_5^2 - 3x_5 + 1, x_5^5 - 3x_5^4 - 6x_5^3 - 19x_5^2 + 9x_5 - 1\}.
\end{aligned}$$

The above process presents the results of the computation with respect to two specific variable orderings, while Table 1 shows the experimental results with respect to two classes of variable orderings with the first two maximal variables being $x_4 > x_5 > \cdots$ and $x_1 > x_2 > \cdots$, respectively. Therein, the "PEO" column shows whether the variable orderings with the corresponding maximal variables in the "Max-Var" column are perfect elimination orderings or not. By "#proj" we denote the sum of the size of those projection polynomial sets obtained in the CAD projection *w.r.t.* a variable ordering. Finally, the "avg" column shows the average of those #proj values with respect to the six orderings corresponding to each pair of maximal variables.

For this example, it is true that the CAD projection tends to generate fewer polynomials when a perfect elimination ordering is used instead of a random one.

Max-Var	PEO	#proj			avg
$x_4 > x_5 > \cdots$	yes	21	20	21	20.7
		21	20	21	
$x_1 > x_2 > \cdots$	no	53	74	34	48.8
		33	52	47	

Table 1: Different variable orderings in Example 4.4.

Definition 4.5. ([5]) Let $A \subset \mathbb{Z}[\bar{x}]$ be a squarefree basis [13] with at least two variables. Brown's projection operator $\mathtt{Proj}_{\mathrm{br}}(\cdot,\cdot)$ is defined as follows: Set $B = \{f \in A \mid x_n \in \mathtt{var}(f)\}$, then

$$\mathtt{Proj}_{\mathrm{br}}(A, x_n) = \mathtt{cont}(A, x_n) \cup \bigcup_{f,g\in B, f\neq g} \{\mathtt{res}(f, g, x_n)\} \cup \bigcup_{f\in B} \{\mathtt{lc}(f, x_n), \mathtt{dis}(f, x_n)\}.$$

Similarly, Prop. 4.2 holds for Brown's operator $\mathtt{Proj}_{\mathrm{br}}$. Also, by Prop. 3.3 and Prop. 4.1, each set in the projection procedure of any $A \subset \mathbb{Z}[\bar{x}]$ obtained by the operator $\mathscr{F}(\mathtt{Proj}_{\mathrm{br}}(\cdot,\cdot))$ preserves the chordal structure of A as well.

Example 4.6. Let

$$F = \{x_1 + x_2 + 2, x_2x_3 + 2x_3 + x_1, x_3x_4 + x_2x_4 + x_3 - 1, x_4 + x_2\}$$

be a squarefree basis. The associated graph $\mathscr{G}(F)$ of F is a chordal graph with $x_1 > x_2 > x_3 > x_4$ a perfect elimination ordering. As in Example 4.4, by successively applying the operator $\mathscr{F}(\mathtt{Proj}_{\mathrm{br}}(\cdot,\cdot))$ in that order, we can finally obtain the full set of projection polynomials: $\{x_4, x_4 + 1, x_4 - 1, x_4 - 2\}$.

If a random projection ordering, say, $x_2 > x_3 > x_4 > x_1$, which is not a perfect elimination ordering, is used in the CAD projection, then we will finally obtain the following polynomial set:

$$\begin{aligned}
\{&x_1, 5 + 4x_1, 9 + 8x_1, 25 + 24x_1, x_1 + 1, x_1 + 2, x_1 + 3, 3x_1 + 4,\\
&2x_1 + 5, 32x_1^2 + 56x_1 + 25, 20x_1^2 + 44x_1 + 25, 13x_1^2 + 34x_1 + 25,\\
&4x_1^3 + 24x_1^2 + 44x_1 + 25, 4x_1^3 - 71x_1^2 - 172x_1 - 100,\\
&x_1^2 + 4x_1 + 5, x_1^4 + 7x_1^3 + 21x_1^2 + 34x_1 + 25\}.
\end{aligned}$$

Max-Var	PEO	#proj			avg
$x_1 > \cdots$	yes	13	13	13	12.7
		13	12	12	
$x_2 > \cdots$	no	26	25	35	33.3
		37	36	41	
$x_3 > \cdots$	no	14	14	22	17.2
		21	15	17	
$x_4 > \cdots$	yes	12	12	19	16.2
		24	14	16	

Table 2: Different variable orderings in Example 4.6.

Moreover, the sum of the size of the projection polynomial sets obtained via each of the 24 variable orderings is shown in Table 2. Again we see that, when the perfect elimination orderings are used, the CAD projection may yield smaller polynomial sets than it does when other kind of variable orderings are used instead.

5 COMPLEXITY ANALYSIS: THE "SIZE" OF THE PROJECTION POLYNOMIALS

In this section, we discuss the complexity of the CAD computation via a perfect elimination ordering by giving a new estimate of the "size" of the full set of projection polynomials obtained via that ordering. The results in this section are obtained by the method of Bradford et al. [4], based on which they estimate the growth of the "size" of the projection polynomials.

For a set of polynomials $A \subset \mathbb{Z}[\bar{x}]$, *the combined degree* [24] of A refers to the number $\max_{x \in \mathtt{var}(A)} \deg(\prod A, x)$, where $\prod A$ is the product of all the polynomials in A. And we say the set A has *the* (m, d)*-property* [4] if it can be partitioned into at most m pairwise disjoint subsets, each with combined degree at most d. This type of properties are useful while tracking the size of the projection polynomials.

The following lemma estimates the growth of the "size" of the projection polynomial sets while using the operator $\mathtt{Proj}_{\mathrm{mc}}$.

Lemma 5.1. *([4], Lemma 11, Corollary 12) Suppose $A \subset \mathbb{R}[\bar{x}]$ is a set of polynomials with the (m, d)-property and $x \in \mathtt{var}(A)$, then the set $\mathtt{Proj}_{\mathrm{mc}}(A, x)$ has the $(M, 2d^2)$-property with $M = \left\lfloor \frac{(m+1)^2}{2} \right\rfloor$. If $m > 1$, then the set $\mathtt{Proj}_{\mathrm{mc}}(A, x)$ has the $(m^2, 2d^2)$-property.*

In Subsection 5.1 we recall the general complexity analysis for CAD while in Subsection 5.2 we provide the complexity analysis with respect to the case where a perfect elimination ordering is used during the CAD projection.

5.1 Complexity Analysis for CAD with General Variable Ordering

In Table 3 ([4] Table 1), Lemma 5.1 is applied recursively to estimate the growth of the pair (m, d) in the (m, d)-property during the CAD projection rendered by the operator $\mathtt{Proj}_{\mathrm{mc}}$. The "Number" column shows the upper bound of the number of the subsets in a possible partition and the "Degree" column shows the upper bound of the corresponding combined degree of those subsets, and $M = \left\lfloor \frac{(m+1)^2}{2} \right\rfloor$.

Table 3: Growth of the "size" of the polynomial sets in CAD projection.

Variables	Number	Degree
n	m	d
$n-1$	M	$2d^2$
$n-2$	M^2	$8d^4$
$\vdots$	$\vdots$	$\vdots$
$n-r$	$M^{2^{r-1}}$	$2^{2^r-1}d^{2^r}$
$\vdots$	$\vdots$	$\vdots$
1	$M^{2^{n-2}}$	$2^{2^{n-1}-1}d^{2^{n-1}}$

The number of the real roots of the product of the polynomials in a univariate polynomial set with the (m, d)-property is at most md. And the number of the corresponding cells in $\mathbb{R}^1$ is twice the number of the real roots plus one. Therefore, the total number of the cells in the CAD of $\mathbb{R}^n$ is at most the product of those $2m'd' + 1$, with m' and d' taking the corresponding values in the "Number" and the "Degree" columns in Table 3, *i.e.*

$$(2md + 1) \prod_{r=1}^{n-1} \left[2(2^{2^r-1} M^{2^{r-1}} d^{2^r}) + 1 \right].$$

So the following estimate is obtained:

Theorem 5.2. *([4]) The number of the CAD cells in $\mathbb{R}^n$ obtained by the operator $\mathtt{Proj}_{\mathrm{mc}}$ is $O((2d)^{2^n-1} M^{2^{n-1}-1} m)$ with $M = \left\lfloor \frac{(m+1)^2}{2} \right\rfloor$.*

5.2 Complexity Analysis for CAD with Perfect Elimination Ordering

First, we introduce the concept of elimination tree to further exploit the relationship between the variables during the perfect elimination process. For convenience, we assume in this section that all the associated graphs of the polynomial sets are connected.

Definition 5.3. ([12] Def. 2.5) Let G be a chordal graph with a perfect elimination ordering $x_1 < x_2 < \ldots < x_n$. *The elimination tree* of G is the following directed spanning tree: for each $l \neq 1$ there is an arc from x_l toward the largest x_p that is adjacent to x_l and $x_p < x_l$. We will say that x_p is the parent of x_l and x_l is a child of x_p (we denote by $\mathtt{child}(x_p)$ the set of all the children of x_p). Note that the elimination tree is rooted at x_1.

Lemma 5.4. *Let T be the elimination tree of a chordal graph G. If $x_s < x_t$ and (x_s, x_t) is an edge of G, then there is a path from x_t to x_s in T.*

Proof. If x_s is the parent of x_t, there is nothing to prove. Otherwise, let $x_{t'}$ be the parent of x_t. By Def. 5.3, $x_s < x_{t'}$. Because (x_s, x_t) and $(x_{t'}, x_t)$ are edges of G which is a chordal graph, $(x_s, x_{t'})$ is also an edge of G. We are done by repeating the same discussion. □

We will see that the path from a node to the root in the elimination tree is exactly the projection path of the polynomials with the variable corresponding to this node being the maximal variable. It is in this way the elimination tree describes the relationships between those variables, and this enables us to give a new complexity analysis.

Let $A \subset \mathbb{Z}[\bar{x}]$ be a polynomial set, G a chordal structure of A with a perfect elimination ordering $x_1 < \ldots < x_n$ and T the elimination tree of G with respect to that ordering. We denote by P_i the projection polynomial sets of the operator $\mathtt{Proj}_{\mathrm{mc}}$, *i.e.*

$$P_n = A \text{ and } P_i = \mathtt{Proj}_{\mathrm{mc}}(P_{i+1}, x_{i+1}) \text{ for } i = n-1, \ldots, 1.$$

We define T_p to be the projection polynomial sets of the operator $\mathtt{Proj}_{\mathrm{mc}}$ along the elimination tree as follows:

Definition 5.5. For each $1 \leq p \leq n$ we define

$$A_p = \{f \in A \mid \max(\mathtt{var}(f)) = x_p\}$$

and

$$T_p = A_p \cup \bigcup_{x_l \in \mathtt{child}(x_p)} \mathtt{Proj}_{\mathrm{mc}}(T_l, x_l). \quad (1)$$

Note that when x_p is a leaf of T, $T_p = A_p$.

Proposition 5.6. *For $i = 1, \ldots, n$, we have*

$$\{f \in T_i \mid x_i \in \mathtt{var}(f)\} = \{f \in P_i \mid x_i \in \mathtt{var}(f)\}.$$

Furthermore, we have

$$\{f \in \bigcup_i T_i \mid \mathtt{var}(f) \neq \emptyset\} = \{f \in \bigcup_i P_i \mid \mathtt{var}(f) \neq \emptyset\}.$$

Proof. It is clear that $\{f \in T_i \mid x_i \in \mathtt{var}(f)\} \subset P_i$. So we only need to prove that for any $1 \le p \le n$,

$$\{f \in P_p \mid x_p \in \mathtt{var}(f)\} \subset T_p. \tag{2}$$

When $p = n$, the claim holds. Assume that the containment (2) holds for $p = n, n-1, \ldots, k+1$. In the following we prove that it also holds for $p = k$. Set $g \in P_k$ to be a polynomial such that $x_k \in \mathtt{var}(g)$ and set $l = \max(\{1 \le i \le n \mid g \in P_i\}) \ge k$. If $l = n$, then $g \in A_k \subset T_k$. Otherwise $l < n$, and $g \notin P_{l+1}$ but $g \in P_l$. Suppose that the polynomial g is constructed from a polynomial $h \in P_{l+1}$ (or from two polynomials $h_1, h_2 \in P_{l+1}$), then we have $x_{l+1} \in \mathtt{var}(h)$ (*resp.*, $x_{l+1} \in \mathtt{var}(h_1) \cap \mathtt{var}(h_2)$). Since $x_k \in \mathtt{var}(g)$, $x_k \in \mathtt{var}(h)$ (*resp.*, $x_k \in \mathtt{var}(h_1) \cup \mathtt{var}(h_2)$). So there is a polynomial (h, h_1 or h_2) that contains both the variables x_k and x_{l+1}. This means that (x_k, x_{l+1}) is an edge of G. By Lemma 5.4, there is a path from x_{l+1} to x_k in T. On the other hand, $l + 1 > k$. So by the inductive assumption, (2) holds for $p = l + 1$ and we have $h \in T_{l+1}$ (*resp.*, $h_1, h_2 \in T_{l+1}$). Thus $g \in \mathtt{Proj}_{\mathrm{mc}}(T_{l+1}, x_{l+1}) \subset T_q$ with q such that x_q is the parent of x_{l+1} in T. Since $\max(\mathtt{var}(g)) = x_k$, g remains unchanged in the projection process along the sub-path from x_q to x_k in T (which is defined by (1)). Therefore, $g \in T_k$.

Further, for every $g \in \bigcup_i P_i$ so that $\mathtt{var}(g) \ne \emptyset$. Set $k = \min(\{1 \le i \le n \mid g \in P_i\})$, then $x_k \in \mathtt{var}(g)$ and $g \in T_k$. Therefore $\{f \in \bigcup_i P_i \mid \mathtt{var}(f) \ne \emptyset\} \subset \{f \in \bigcup_i T_i \mid \mathtt{var}(f) \ne \emptyset\}$. The reverse containment can be proved similarly. □

The above proposition shows that the difference between $\cup T_i$ and $\cup P_i$ merely consists of some constants, which make no differences in the CAD computation. So we can estimate the growth of the "size" of the projection polynomials by estimating the "size" of T_i.

Lemma 5.7. *For a variable x_p, if the set A_p has the (m, d)-property and the set T_l also has this property for each $x_l \in \mathtt{child}(x_p)$, then T_p has the $\left((|\mathtt{child}(x_p)| + 1)M, 2d^2\right)$-property with $M = \left\lfloor \frac{(m+1)^2}{2} \right\rfloor$. When $m > 1$, the set T_p has the $\left((|\mathtt{child}(x_p)|+1)m^2, 2d^2\right)$-property.*

Proof. By Def. 5.5, $T_p = A_p \cup \bigcup_{x_l \in \mathtt{child}(x_p)} \mathtt{Proj}_{\mathrm{mc}}(T_l, x_l)$. For any $x_l \in \mathtt{child}(x_p)$, the set $\mathtt{Proj}_{\mathrm{mc}}(T_l, x_l)$ has the $(M, 2d^2)$-property with $M = \left\lfloor \frac{(m+1)^2}{2} \right\rfloor$ by Lemma 5.1. And A_p has the (m, d)-property. Since $m \le M$ and $d < 2d^2$, A_p also has the $(M, 2d^2)$-property. □

Table 4: Growth of the "size" of the polynomial sets in CAD projection with perfect elimination ordering.

Height	Number	Degree
0	m	d
1	$(w+1)M$	$2d^2$
2	$(w+1)^3M^2$	$8d^4$
3	$(w+1)^7M^4$	$128d^8$
$\vdots$	$\vdots$	$\vdots$
r	$(w+1)^{2^r-1}M^{2^{r-1}}$	$2^{2^r-1}d^{2^r}$
$\vdots$	$\vdots$	$\vdots$
h	$(w+1)^{2^h-1}M^{2^{h-1}}$	$2^{2^h-1}d^{2^h}$

As in Section 5.1, we apply Lemma 5.7 recursively to estimate the growth of the "size" of the sets T_i. The results are shown in Table 4 with $M = \left\lfloor \frac{(m+1)^2}{2} \right\rfloor$ and $w = \max_{1 \le l \le n}(|\mathtt{child}(x_l)|)$ therein. This is how one reads the table: a triple (h', m', d') shown in a row means that for any node x_i with height h' in T, the set T_i has the (m', d')-property. The main difference of this table from Table 3 is that we replace the number of variables therein by the height of the nodes in the perfect elimination tree. Rigorously, we have

Theorem 5.8. *Suppose the set A_l has the (m, d)-property for any $1 \le l \le n$. Then for every internal node x_i with height h_i, the set T_i has the*

$$\left((w+1)^{2^{h_i}-1}M^{2^{h_i-1}}, 2^{2^{h_i}-1}d^{2^{h_i}}\right)\text{-property}$$

with $w = \max_{1 \le l \le n}(|\mathtt{child}(x_l)|)$ and $M = \left\lfloor \frac{(m+1)^2}{2} \right\rfloor$. Note that for a leaf node x_l, the set $T_l = A_l$ has the (m, d)-property by assumption.

The following theorem gives the upper bound of the number of the CAD cells according to Theorem 5.8 and Prop. 5.6.

Theorem 5.9. *If the set A_l has the (m, d)-property for every x_l, then the number of CAD cells in $\mathbb{R}^n$ is at most $\prod_{i=1}^n (2K_i + 1)$, where*

$$K_i = \begin{cases} md, & \text{if } x_i \text{ is a leaf node} \\ \left(2(w+1)\right)^{2^{h_i}-1}M^{2^{h_i-1}}d^{2^{h_i}}, & \text{otherwise} \end{cases}$$

with $w = \max_{1 \le l \le n}(|\mathtt{child}(x_l)|)$, $M = \left\lfloor \frac{(m+1)^2}{2} \right\rfloor$ and h_i the height of the node x_i in the tree T.

Comparing Table 4 with Table 3, we can see some significant differences. The first one is that the (m, d)-property is for A in Table 3, but it is for the subsets $A_l \subset A$ in Table 4. This means Table 4 (hence Theorem 5.9) may accept a smaller pair (m, d) than Table 3 (*resp.* Theorem 5.2) does. The second difference is the quantity in the double-exponent: one is the number of variables while the other is the height of the node. In general, the difference between these two is not big. But in the case of sparsity, the latter estimate will be much smaller than the former one. This is consistent with our guess at the very beginning. Also, there is a double exponent with $w + 1 \ge 2$ as the base in the new estimate, but note that w describes the number of branches and the height is lower if there are more branches, and vice versa. So this won't be too bad for the new estimate.

6 EXPERIMENTS

In this section, we compare the performance of the CAD computation via perfect elimination orderings and via other random variable orderings by testing some polynomial sets with variable sparsity. As explained before, computing CAD projection via a perfect elimination ordering preserves the chordal structure (as well as the pattern of sparsity embedded in it). This is how we exploit the variable sparsity in the CAD computation. And because of this, the CAD computation via a perfect elimination ordering usually performs better on the examples with variable sparsity.

All tests were conducted on 6-Core Intel Core i7-8750H@2.20GHz with 32GB of memory and ARCH LINUX SYSTEM.

6.1 The Impact of the Tree Height

According to the result in Section 5, the new complexity analysis for the CAD computation via a perfect elimination ordering has two main differences from the original one for the CAD computation via a general variable ordering: one is the new (m, d)-property and the other is the height of the elimination tree introduced in the formulae. Among them, the impact of the tree height on the new complexity is particularly noteworthy, and the height of the elimination trees with respect to different perfect elimination orderings of the same chordal structure can be different. We recommend not only the perfect elimination orderings as the variable orderings used in the CAD projection, but also the ones among them which result in perfect elimination trees of the minimal height. The following example shows how this can be used to reduce the computation.

Example 6.1. The following polynomial set forms a lattice reachablility problem which is described in [15]:

$$F_n = \{x_k x_{k+3} - x_{k+1}x_{k+2} \mid k = 1, 2, \ldots, n-3\}, 4 \le n \in \mathbb{Z}.$$

The associated graph $\mathscr{G}(F_n)$ of the set F_n is shown in Fig. 4 and it is obviously already a chordal structure of F_n.

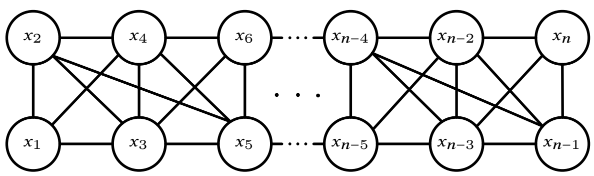

Figure 4: Associated graph $\mathscr{G}(F_n)$ of F_n in Example 6.1.

We consider two different perfect elimination orderings

$$l_n^1 : x_1 > x_2 > x_3 > \ldots > x_{n-1} > x_n, \text{ and}$$
$$l_n^2 : x_1 > \cdots > x_{\lceil \frac{n-3}{2} \rceil} > x_n > \ldots > x_{\lceil \frac{n-1}{2} \rceil}.$$

The figures below (Fig. 6 and Fig. 7) show the elimination trees with respect to the ordering l_n^1 and l_n^2, respectively.

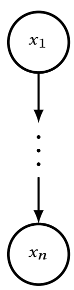

Figure 6: The elimination tree with respect to l_n^1.

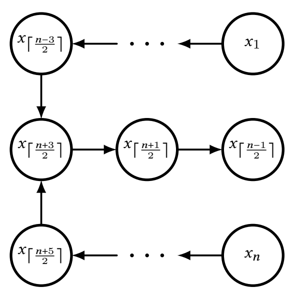

Figure 7: The elimination tree with respect to l_n^2.

Note that the height of the tree in Fig. 6 is $n-1$ and the height of the tree in Fig. 7 is $\lceil \frac{n+1}{2} \rceil$. Hence, from the complexity analysis in the last section, l_n^2 can be a better variable ordering. We compute the CAD's for the polynomial set F_n by the CAD tools in Mathematica 12 and in Maple 2020. In Mathematica 12, we use the command "CylindricalDecomposition" [30] for the formula $\bigwedge_{f \in F_n} f \neq 0$ while in Maple 2020, we use the command "PartialCylindricalAlgebraicDecomposition" [32] for the polynomial $\prod F_n$ instead. The results are shown in Table 5. We see that the runtime with respect to the ordering l_n^2 with a smaller elimination tree height is much shorter. Also, less sample points are obtained with respect to the ordering l_n^2 (For this example, these two tools give the same number of sample points in each case shown in the table). Both of these are well consistent with the previous complexity analysis.

Table 5: Comparing the runtime and the number of sample points of the CAD computation via the perfect elimination orderings l_n^1 and l_n^2 with different elimination tree height ("OM" means out of memory).

n		l_n^1	l_n^2	Random Orderings		
8	Mathematica 12	0.32s	0.17s	18.13s	28.17s	1.87s
	Maple 2020	0.09s	0.04s	8.80s	5.97s	0.27s
	#sample	576	256	11520	16000	1728
9	Mathematica 12	1.04s	0.33s	7.95s	4.66s	43.16s
	Maple 2020	0.28s	0.08s	1.32s	1.0s	24.86s
	#sample	1728	512	5184	3456	23552
13	Mathematica 12	122s	22.4s	>2h	>2h	>2h
	Maple 2020	397s	29.1s	>2h	OM	OM
	#sample	139968	41472	–	–	–
14	Mathematica 12	405s	74.3s	>2h	>2h	>2h
	Maple 2020	1744s	329s	>2h	OM	>2h
	#sample	419904	124416	–	–	–

6.2 The Impact of Different Chordal Completions

In practice, the associated graph of a sparse polynomial system is not necessarily a chordal graph. In this case we need to compute one of its chordal completions together with a perfect elimination ordering of that completion. Conversely, from any variable ordering one can also construct a corresponding chordal completion via the elimination game. For the case where the associated graph is not chordal, we recommend those variable orderings for CAD computation, which, via the elimination game, result in minimal chordal completions. The following example shows the impact of different chordal completions on the CAD computation.

Example 6.2. ([12]) Consider the following polynomial set:

$$I^{n_1,n_2} = \bigcup_{0 \le i < n_1, 0 \le j < n_2} \left\{ \begin{array}{c} U_{i,j}R_{i,j+1} - R_{i,j}U_{i+1,j}, \\ D_{i,j+1}R_{i,j} - R_{i,j+1}D_{i+1,j+1}, \\ D_{i+1,j+1}L_{i+1,j} - L_{i+1,j+1}D_{i,j+1}, \\ U_{i+1,j}L_{i+1,j+1} - L_{i+1,j}U_{i,j} \end{array} \right\}.$$

In Table 6 we show the runtime and the numbers of sample points of the CAD computation for I^{n_1,n_2}. Therein, the symbol "n" denotes the number of variables and "l_p" denotes the variable orderings which, via the elimination game, result in minimal chordal completions of the associated graph of I^{n_1,n_2}. We know that each of these variable orderings is a perfect elimination ordering of the corresponding completion. The symbol "l_{sv}" denotes the variable orderings given by the Maple command "SuggestVariableOrder" while "l_R" denotes some of the random variable orderings. The letter "t" denotes the runtime of the command "PartialCylindricalAlgebraicDecomposition" for the polynomial $\prod I^{n_1,n_2}$ in Maple 2020, while "m" denotes the number of sample points. The number d is defined by $d = 1 - |G|/|\bar{G}|$ with $|G|$ the number of the edges of the associated graph and $|\bar{G}|$ the number of the edges of its chordal completion obtained by playing the elimination game via the corresponding variable ordering in the table. The number d characterizes the difference between the associated graph and its chordal completion.

Table 6: The CAD computation via different variable orderings resulting different chordal completions.

(n_1, n_2)	n		l_p	l_{sv}	l_R	
(1,1)	8	t	0.04s	0.14s	0.18s	0.12s
		d	0.17	0.17	0.23	0.20
		m	288	864	1153	864
(2,1)	14	t	361s	780s	OM	1132s
		d	0.17	0.21	0.50	0.35
		m	248832	248832	–	248832
(1,2)	14	t	581s	712s	1147s	2868s
		d	0.17	0.14	0.46	0.51
		m	248832	248832	248832	248832

The data in Table 6 show that, roughly speaking, the smaller the number d is, the faster the computation may be done. In other words, a variable ordering could be better for the CAD projection if the difference between the corresponding chordal completion and the original associated graph is smaller. On the other hand, the relations between the number of sample points and d seem not easy to grasp. One thing is clear that, for this example, the numbers of sample points obtained in the cases with smaller d are less than or equal to the ones obtained in the other cases with bigger d. A somewhat surprising fact is that, when $n = 14$, the number of sample points keeps unchanged under several different variable orderings.

7 CONCLUSION

In this paper, we indicate for the first time that classical CAD projection algorithms preserve the chordal structure of a polynomial set, if the projection process is done in accordance with a perfect elimination ordering. More importantly, we provide a complexity analysis for the CAD algorithms executed with respect to a perfect elimination ordering by giving a new (m, d)-property which characterizes the "size" of the corresponding full set of projection polynomials. Combining these theoretical results with the experiments, we find that: (i) When the polynomial set is sparse and of the chordal structure, it can be better to compute the CAD via the perfect elimination orderings which result in perfect elimination trees of the minimal height. (ii) When the polynomial set is not of the chordal structure but still nearly chordal (and also sparse), it could be better to use the variable orderings that result in minimal chordal completions of the original system. The core idea is that we use the perfect elimination ordering during the computation so that the sparsity embedded in the (nearly) chordal structure can pass from the original polynomial set to all the successive polynomial sets produced later on. Both the complexity analysis and the experimental results show that this idea really helps reduce the size of the full set of projection polynomials in many cases.

ACKNOWLEDGMENTS

The work has been supported by the NSFC under grant No. 61732001.

REFERENCES

[1] A. Berry, J. R. S. Blair, P. Heggernes, and B. W. Peyton. 2004. Maximum Cardinality Search for Computing Minimal Triangulations of Graphs. *Algorithmica* 39, 4 (2004), 287–298.

[2] J. R. S. Blair and B. Peyton. 1993. An Introduction to Chordal Graphs and Clique Trees. In *Graph Theory and Sparse Matrix Computation*. Springer New York, 1–29.

[3] R. Bradford, J.H. Davenport, M. England, and D. Wilson. 2013. Optimising Problem Formulation for Cylindrical Algebraic Decomposition. In *International Conference on Intelligent Computer Mathematics*. Springer, 19–34.

[4] R. J. Bradford, J. H. Davenport, M. England, S. McCallum, and D. J. Wilson. 2016. Truth Table Invariant Cylindrical Algebraic Decomposition. *Journal of Symbolic Computation* 76 (2016), 1–35.

[5] C. W. Brown. 2001. Improved Projection for Cylindrical Algebraic Decomposition. *Journal of Symbolic Computation* 32, 5 (2001), 447–465.

[6] C. W. Brown. 2004. Companion to the Tutorial: Cylindrical Algebraic Decomposition. *Presented at ISSAC'04* (2004).

[7] C. W. Brown. 2015. Open Non-Uniform Cylindrical Algebraic Decompositions. In *Proc. ISSAC'15*. ACM Press, 85–92.

[8] C. Chen. 2020. Chordality Preserving Incremental Triangular Decomposition and Its Implementation. In *ICMS'20*. Springer, 27–36.

[9] C. Chen, M. Moreno Maza, B. Xia, and L. Yang. 2009. Computing Cylindrical Algebraic Decomposition via Triangular Decomposition. In *Proc. ISSAC'09*. ACM Press, 95–102.

[10] C. Chen, Z. Zhu, and H. Chi. 2020. Variable Ordering Selection for Cylindrical Algebraic Decomposition with Artificial Neural Networks. In *International Congress on Mathematical Software*. Springer, 281–291.

[11] D. Cifuentes and P. A. Parrilo. 2016. Exploiting Chordal Structure in Polynomial Ideals: A Gröbner Bases Approach. *SIAM Journal on Discrete Mathematics* 30, 3 (2016), 1534–1570.

[12] D. Cifuentes and P. A. Parrilo. 2017. Chordal Networks of Polynomial Ideals. *SIAM Journal on Applied Algebra and Geometry* 1, 1 (2017), 73–110.

[13] G. E. Collins. 1975. Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition. In *Automata Theory and Formal Languages, 2nd GI Conference, Kaiserslautern, May 20-23, 1975 (Lecture Notes in Computer Science, Vol. 33)*, H. Barkhage (Ed.). Springer, 134–183.

[14] G. E. Collins and H. Hong. 1991. Partial Cylindrical Algebraic Decomposition for Quantifier Elimination. *Journal of Symbolic Computation* 12, 3 (1991), 299–328.

[15] P. Diaconis, D. Eisenbud, and B. Sturmfels. 1998. *Lattice Walks and Primary Decomposition*. Birkhäuser Boston, 173–193.

[16] A. Dolzmann, A. Seidl, and T. Sturm. 2004. Efficient Projection Orders for CAD. In *Proc. ISSAC'04*. ACM Press, 111–118.

[17] M. England and D. Florescu. 2019. Comparing Machine Learning Models to Choose the Variable Ordering for Cylindrical Algebraic Decomposition. In *International Conference on Intelligent Computer Mathematics*. Springer, 93–108.

[18] J. Han, L. Dai, H. Hong, and B. Xia. 2017. Open Weak CAD and Its Applications. *Journal of Symbolic Computation* 80 (2017), 785–816.

[19] J. Han, Z. Jin, and B. Xia. 2016. Proving Inequalities and Solving Global Optimization Problems via Simplified CAD Projection. *Journal of Symbolic Computation* 72 (2016), 206–230.

[20] H. Hong. 1990. An Improvement of the Projection Operator in Cylindrical Algebraic Decomposition. In *Proc. ISSAC'1990*. ACM Press, 261–264.

[21] H. Hong. 1990. *Improvements in CAD-Based Quantifier Elimination*. Ph.D. Dissertation. The Ohio State University.

[22] H. Hong and M. Safey El Din. 2012. Variant Quantifier Elimination. *Journal of Symbolic Computation* 47, 7 (2012), 883–901.

[23] Z. Huang, M. England, D. Wilson, J. H. Davenport, L.C. Paulson, and J. Bridge. 2014. Applying Machine Learning to the Problem of Choosing a Heuristic to Select the Variable Ordering for Cylindrical Algebraic Decomposition. In *International Conference on Intelligent Computer Mathematics*. Springer, 92–107.

[24] S. McCallum. 1985. *An Improved Projection Operation for Cylindrical Algebraic Decomposition*. Ph.D. Dissertation. University of Wisconsin-Madison.

[25] S. McCallum. 1988. An Improved Projection Operation for Cylindrical Algebraic Decomposition of Three-Dimensional Space. *Journal Symbolic Computation* 5, 1/2 (1988), 141–161.

[26] S. McCallum. 1998. An Improved Projection Operation for Cylindrical Algebraic Decomposition. In *Quantifier Elimination and Cylindrical Algebraic Decomposition*. Springer, 242–268.

[27] C. Mou, Y. Bai, and J. Lai. 2019. Chordal Graphs in Triangular Decomposition in Top-Down Style. *Journal of Symbolic Computation* 102 (2019), 108–131.

[28] C. Mou and J. Lai. 2019. On the Chordality of Simple Decomposition in Top-Down Style. In *International Conference on Mathematical Aspects of Computer and Information Sciences*. Springer, 138–152.

[29] S. Parter. 1961. The Use of Linear Graphs in Gauss Elimination. *SIAM Review* 3, 2 (1961), 119–130.

[30] A. W. Strzeboński. 2006. Cylindrical Algebraic Decomposition Using Validated NumericS. *Journal of Symbolic Computation* 41, 9 (2006), 1021–1038.

[31] A. W. Strzeboński. 2016. Cylindrical Algebraic Decomposition Using Local Projections. *Journal of Symbolic Computation* 76 (2016), 36–64.

[32] L. Yang, X. Hou, and B. Xia. 2001. A Complete Algorithm for Automated Discovering of a Class of Inequality-Type Theorems. *Science in China Series F Information Sciences* 44, 1 (2001), 33–49.