TECHNIQUES FOR RITT-WU’S DECOMPOSITION ALGORITHM

Shang-Ching Chou and Xiao-Shan Gao

Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712-1188

TR-90-02 February 1990
Techniques for Ritt–Wu’s Decomposition Algorithm*

Shang-Ching Chou and Xiao-Shan Gao†
Department of Computer Sciences
The University of Texas at Austin, Austin, Texas 78712 USA

Abstract This is a summary of techniques used in improving Ritt–Wu’s decomposition algorithm. Some of them are reported here for the first time, although they have been extensively used by the authors in the connection with geometric reasoning. The algorithm is to decompose an algebraic set into a union of its irreducible varieties. With constant efforts by many researchers, the current techniques can speed up the algorithm by a magnitude of two orders. The detailed data of eight examples, including the Morley configuration and the $S_3$ configuration, are collected in the Appendix.

Keywords Polynomial, ideal, prime ideal, algebraic set, irreducible variety, ascending chain, characteristic set, pseudo division, Ritt–Wu’s principle, Gröbner basis, decomposition of an algebraic set, irreduntant decomposition.

* The work reported here was supported in part by the NSF Grant CCR-8702108.
† On leave from Institute of Systems Science, Academia Sinica, Beijing.
1. Introduction

Algorithms for decomposition of an algebraic set into a union of its irreducible varieties have been known since the work of G. Hermann [8]. However, Ritt's decomposition algorithm has been recognized and revived only recently by Wu's work [14], [15]. Wu has added many new, important ideas to Ritt's original algorithm; so it is proper to call the algorithm Ritt-Wu's decomposition algorithm. In order to make the algorithm efficient and practical, many things need to be done. Since Ritt-Wu's algorithm is highly undeterministic, it provides many variants and combinations of variants. Some variants can lead to essential improvements of the algorithm. Many researchers have been studied and experimented with various variants since 1983 [14], [1], [9], [15], [19], [17], [16], [10], [3], [18], [13].

This report is a summary of techniques used to improve the decomposition algorithm. Many of them, especially the technique in 4.3.3, are reported in this paper for the first time, although they have been extensively used by the authors in the connection with mechanical geometry reasoning. It is for those who have certain prior knowledge about Ritt-Wu's work and want to study and experiment with the decomposition algorithm seriously. Thus we will use some notations or terminologies without detailed explanations. The reader can found detailed presentation in [14], [3], [2].

2. Basic Terminologies

Let $K$ be a computable field such as $\mathbb{Q}$, the field of rational numbers, and $y = y_1, y_2, \ldots, y_m$ be indeterminates. Unless stated otherwise, all polynomials mentioned in this section are in $K[y_1, \ldots, y_m] = K[y]$.

Let $f \in K[y]$. The class of $f$, denoted by $\text{class}(f)$, is the largest $i$ such that $y_i$ occurs in $f$. If $f \in K$, then $\text{class}(f) = 0$. Let $c = \text{class}(f) > 0$. We call $y_c$, denoted by $\text{l}(f)$, the leading variable of $f$. Considering $f$ as a polynomial in $y_c$, we can write $f$ as

$$a_n y_c^n + a_{n-1} y_c^{n-1} + \cdots + a_0$$

where $a_n, \ldots, a_0$ are in $K[y_1, \ldots, y_{c-1}]$, $n > 0$, and $a_n \neq 0$. We call $a_n$ the initial or leading coefficient of $f$ and $n$ the leading degree of $f$, denoting them as $\text{lc}(f)$ and $\text{ld}(f)$, respectively. Pseudo remainder of $g$ by $f$ (in the variable $y_c$) is denoted by $\text{prem}(g, f)$.

**Definition (2.1).** Let $C = f_1, f_2, \ldots, f_r$ be a sequence of polynomials in $K[y]$. We call it a quasi ascending chain (abbrev. quasi asc chain) or a triangular form if either $r = 1$ and $f_1 \neq 0$, or $r > 1$ and $0 < \text{class}(f_1) < \text{class}(f_2) < \cdots < \text{class}(f_r)$.

Unless stated otherwise, we assume $C$ is non-trivial, i.e., $\text{class}(f_1) > 0$. Let $f_1, \ldots, f_r$ be a quasi asc chain. We define $\text{prem}(g, f_1, \ldots, f_r)$ inductively to be $\text{prem}(\text{prem}(g, f_r), f_1, \ldots, f_{r-1})$. Let it be $R$. Then we have the following important **Remainder Formula**:

$$(2.1.1) \quad I^+_1 \cdots I^+_r g = Q_1 f_1 + \cdots + Q_r f_r + R$$

where the $I_i = \text{lc}(f_i)$, $s_1, \ldots, s_r$ are some non-negative integers, $Q_1, \ldots, Q_r$ are polynomials. Furthermore, $\text{deg}(R, x_i) < \text{deg}(f_i, x_i)$, for $i = 1, \ldots, r$, where $x_i = \text{l}(f_i)$.

(i) A quasi ascending chain is called an ascending chain in weak sense (abbrev. w-asc chain) if $\text{prem}(\text{l}(f_i), f_1, \ldots, f_r) \neq 0$, for $i = 1, \ldots, r$. 
(ii) A w-asc chain is called an *ascending chain in Wu’s sense* (abb. wu-asc chain) if the 
\(\deg(lc(f_j), lv(f_i)) < ld(f_i)\) for \(i < j\).

(iii) A wu-asc chain is called an *ascending chain in Ritt’s sense* (abb. r-asc chain) if \(\deg(f_j, lv(f_i)) < ld(f_i)\) for \(i < j\).

Whenever we talk about an asc chain, it can be one of the about three.

We define a partial order \(<\) in \(K[y]: f < g\) if \(\text{class}(f) < \text{class}(g)\) or \(\text{class}(f) = \text{class}(g) > 0\) and \(ld(f) < ld(g)\). If neither \(f < g\) nor \(g < f\), we denote \(f \sim g\). Obviously, this partial order is well founded, i.e., there is no infinite, strictly decreasing sequences of polynomials.

**Definition (2.2).** Let \(C = f_1, ..., f_r\) and \(C_1 = g_1, ..., g_m\) be two ascending chains. We define \(C < C_1\) if there is an \(s\) such that \(s \leq \min(r, m)\) and \(f_i\) and \(g_i\) are of the same rank for \(i < s\) and that \(f_s < g_s\), or \(m < r\) and \(f_i \sim g_i\) for \(i \leq m\).

**Proposition (2.3).** The partial order \(<\) among the set of all ascending chains is well-founded, i.e., there are no infinite, strictly decreasing sequences of asc chains.

*Proof.* See Lemma 1 of [14].

**Definition (2.4).** Let \(S\) be a nonempty polynomial set. A minimal ascending chain in the set of all chains formed from polynomials in \(S\) is called a *basic set* of \(S\).

Unless stated otherwise, whenever we talk about a finite polynomial set \(S\), we assume \(S\) does not contain zero. By (2.3), every nonempty polynomial set \(S\) has a basic set. Actually, we have the following

**Algorithm (2.5).** Let \(S\) be a finite, non-empty polynomial set. The algorithm is to construct a basic set of \(S\).

*Proof.* See [3].

3. The Decomposition Algorithm

Now let us fix an extension \(E\) of the base field \(K\). Let \(PS\) be a finite polynomial set. We denote \(\text{Zero}(PS)\) the common zeros of \(PS\) in \(E^m\), i.e., the set

\[
\{(a_1, ..., a_m) \in E^m \mid h(a_1, ..., a_m) = 0, \text{ for all } h \in PS\}.
\]

The decomposition algorithm is to decompose \(\text{Zero}(PS)\). It works for any extension \(E\) of \(K\), but is complete only for the case when \(E\) is algebraically closed. Thus, in what follows, we assume \(E\) to be algebraically closed. Then the decomposition algorithm is to decompose \(\text{Zero}(PS)\) into the union of its irreducible varieties.

The algorithm consists of two phases. Phase 1 is to triangulate a polynomial set (initially the set \(PS\)) to obtain an asc chain and other polynomial sets to be decomposed further. This phase is usually called Ritt–Wu’s principle. Phase 2 is to check whether that asc chain is irreducible, if not, then split the polynomial set to be decomposed into two or more polynomial sets. Recursively repeat this process for each of the new polynomial sets until no new polynomial sets are produced. We finally have a set of irreducible asc chains \(ASC_1, ..., ASC_l\) and the decomposition (if \(l > 0\),...
then \( \text{Zero}(PS) \) is empty) desired:

\[
(3.0.1) \quad \text{Zero}(PS) = \bigcup_{i=1}^{l} \text{Zero}(PD(ASC_i)).
\]

where the \( PD(ASC_i) = \{ g \mid \text{prem}(g, ASC_i) = 0 \} \) are prime ideals.

Let \( DS \) be another polynomial set. A key notation introduced by Wu is \( \text{Zero}(PS/DS) \), which is defined to be the set difference \( \text{Zero}(PS) - \text{Zero}(DS) \). As we will see, this notation will lead to essential improvement of the decomposition algorithm.

3.1. Phase 1 (Ritt–Wu’s principle) Let \( PS \) be as before. The algorithm is to construct an ascending chain \( ASC \) such that either

(3.1.1). \( ASC \) consists of a non-zero constant in \( K \cap \text{Ideal}(PS) \). In this case, \( \text{Zero}(PS) \) is empty; or

(3.1.2). \( ASC = f_1, \ldots, f_r \) with \( \text{class}(f_1) > 0 \) and such that \( f_i \in \text{Ideal}(PS) \) and \( \text{prem}(p, f_1, \ldots, f_r) = 0 \) for all \( i = 1, \ldots, r \) and \( p \in PS \).

In that case of (3.1.2) we have:

\[
(3.1.3) \quad \text{Zero}(PS) = \text{Zero}(ASC/\{lc(f_1), \ldots, lc(f_r)\}) \cup \bigcup_{i=1}^{r} \text{Zero}(PS \cup \{lc(f_i)\}),
\]

\[
(3.1.4) \quad \text{Zero}(PS) = \text{Zero}(PD(ASC)) \cup \bigcup_{i=1}^{r} \text{Zero}(PS \cup \{lc(f_i)\}).
\]

**Proof.** By (2.5), we can construct a basic set \( BS_1 \) of \( PS_1 = PS \). If \( BS_1 \) consists of only one nonzero constant, then we have (3.1.1). Otherwise, we can expand \( PS_1 \) to \( PS_2 \) by adding nonzero \( \text{prem}(g, BS_1) \) of all \( g \) elements of \( PS_1 \). If \( PS_2 = PS_1 \), then we have (3.1.2). Otherwise, we can construct a basic set \( BS_2 \) of \( PS_2 \). By (2.8), \( BS_1 > BS_2 \). If \( BS_2 \) does not consist of one nonzero constant, then we can expand \( PS_2 \) to \( PS_3 \) using the same procedure. Thus we have a strictly increasing sequence of polynomial sets:

\[ PS_1 \subset PS_2 \subset \cdots, \]

with a strictly decreasing sequence of characteristic sets

\[ BS_1 > BS_2 > \cdots. \]

By (2.3), this decreasing sequence can be only finite. Thus, there is an integer \( k \geq 1 \) such that either \( BS_k \) consists of a nonzero constant or \( PS_k = PS_{k+1} \); then we have either (3.1.1) or (3.1.2), respectively. Formulae (3.1.3) and (3.1.4) are the direct consequences of the Remainder Formula (2.1.1) and (2.1.2).

The asc chain \( ASC \) in (3.1.3) is called a characteristic set of \( PS \) and is denoted by CharSet(PS). The number of all characteristic sets produced in this Phase during the decomposition is denoted by ch-number.

3.2. Phase 2. (Check of Irreducibility). If \( ASC = f_1, \ldots, f_r \) obtained in Phase 1 is irreducible. Then \( PD(ASC) \) is a prime ideal, we have obtained an irreducible variety \( \text{Zero}(PD(ASC)) \)
contained in \( \text{Zero}(PS) \). Otherwise, we can use factorization to find two polynomials \( g_1 \) and \( g_2 \) reduced wrpt \( f_1, ..., f_r \) such that \( g_1 g_2 \in \text{Ideal}(PS) \). We have
\[
\text{Zero}(PS) = \text{Zero}(PS \cup \{g_1\}) \cup \text{Zero}(PS \cup \{g_2\}).
\]

3.3. The Decomposition Algorithm. The decomposition algorithm recursively uses Phase 1, then Phase 2 for each new polynomial set produced in these two phases. Since for each of the new polynomial sets the asc chain \( ASC_i \) obtained in Phase 1 is lower than its predecessor, The procedure terminates. Upon termination, we either find \( \text{Zero}(PS) \) is empty or have a set of irreducible asc chains \( ASC_1, ..., ASC_i \) that satisfy (3.0.1).

The above algorithm can have the following two major variants.

3.4. The Coarse Form of the Algorithm

This variant, proposed by Wu in [15], does not require that the asc chains \( ASC_i \) in (3.0.1) be irreducible. We only need to drop Phase 2 in algorithm 3.3. The procedure still terminates. The advantage of this variant is that factorization (especially factorization over extension fields) is not needed. The disadvantage is that each \( PD(ASC_i) \) is not necessarily a prime ideal. Application methods based on this variant are incomplete. Avoiding factorization is not necessarily good for the efficiency of the algorithm. Actually, we need factorization (not over extension) to reduce the sizes and degrees of polynomials produced. We will discuss this issue in 4.2.2.

3.5. Decomposition of \( \text{Zero}(PS/DS) \)

Decomposition of \( \text{Zero}(PS/DS) \) is an important variant proposed by Wu. It not only has important applications (e.g., in solving a system of polynomial equations) but also can be used as an important means to control branching in Ritt’s original algorithm (see 4.3.3).

This variant works as follows. When Phase 1 produces characteristic set \( ASC \) we check whether \( \text{prem}(d, ASC) = 0 \) for some \( d \in DS \). If it is, then \( \text{Zero}(PD(ASC_i)/DS) \) is empty, and we can delete those \( ASC_i \) in (3.1.1) and finally have the decomposition:

\[
(3.5.1) \quad \text{Zero}(PS/DS) = \bigcup_{i=1}^{t'} \text{Zero}(PD(ASC_i)/DS).
\]

More important, when a characteristic set \( ASC \) of a polynomial set \( PS \) is produced, we can check whether some \( d \in DS \) is reduced to zero by \( PS \cup ASC \) using some other reductions (e.g., the reduction used in the Gröbner basis method). If it is, then \( \text{Zero}(PS/DS) \) is empty. This is one of the most important means to control branching, especially such situations happen at early stages.

4. Various Techniques for Improvements

4.1. Main Problems in the Efficiency

One can get trouble immediately if implementing the algorithm literally following the description in Section 3. In the earlier experiments in 1985-1986 by Ko and Chou, it was observed that
it took hours to decompose a simple polynomial set. The main problems in the algorithm are: (1) The size growth of polynomials in Phase 1; (2) The larger number of branches in Phases 1 and 2. The authors have made extensively experience since 1985 and accumulated a large number of examples.

Example (4.1). (Feuerbach's Theorem) Let \( PS \) be the set in Example (A.6) in the Appendix. It took 131.5 sec to get Char-Set\((PS)\), whose largest polynomial has 168 terms.

Example (4.2). (Pappus' Theorem) Decomposition of \( PS \) in Example (A.3) produced more than 10,000 characteristic sets in Phase 1. But only about 20 are what we were looked for.

4.2. The Size Control

4.2.1. Use \( w-\text{asc} \) Chains and \( W-\text{prem} \)

To reduce the size growth, Wu introduced \( w-\text{asc} \) chains. However, \( w-\text{asc} \) chains still cannot prevent the size growth in many cases. Example (4.1) was actually computed using \( w-\text{asc} \) chains. However, if we use \( w-\text{asc} \) chains and \( W-\text{prem} \) [3], it only took 1.9s to get \( w-\text{Char-set}(PS) \). It is \( \text{ASC}_1 \) in Example (A.6). Using \( w-\text{asc} \) chains with \( W-\text{prem} \) is one of the most important means to control polynomial sizes in Phase 1. For detailed description of this variant, see [3].

4.2.2. Factorization

If \( PS \) contains a polynomial which is a product of two polynomials \( g = g_1 g_2 \), then \( \text{Zero}(PS) = \text{Zero}(PS \cup \{g_1\}) \cup \text{Zero}(PS \cup \{g_2\}) \). Polynomials \( g_1 \) and \( g_2 \) have lower degrees than that of \( g \), and generally have smaller sizes than that of \( g \). Each \( \text{Zero}(PS \cup \{g_i\}) \) is generally easy to decompose than \( \text{Zero}(PS) \).

The main decision is when to check irreducibility. We can check reducibility each time when a new polynomial is produced. This is certainly safest strategy in the sense that if this strategy cannot overcome the large polynomial size trouble, than the other strategy most likely can also not overcome the same trouble. Though factorization of multivariate polynomials are fast in many current computer algebra systems due to the excellent work by Paul S. Wang [11], [12], it is still time-consuming using this strategy because thousands of thousands new polynomials will be produced in the decomposition algorithm.

(2) The first author uses a strategy very near the spirit of Ritt's original algorithm. Whenever a (weak) characteristic set \( f_1, \ldots, f_r \) is produced in Phase 1, the irreducibility of \( g = \text{prem}(f_i, f_1, \ldots, f_{i-1}) \) (in the variable \( lv(f_i) \)) is checked. If it is, then put all factors of \( g \) back into \( PS \). The reducibility over extension fields is checked only after we obtain the decomposition of the forms (3.0.1) or (3.3.5). This strategy with \( W-\text{prem} \) and the branching control in 4.3 has been used to prove at least 500 geometry theorems according to a formulation that non-degenerate conditions are explicitly given in a geometry statement [3].

(3) The second author uses a strategy which checks reducibility of \( W-\text{prem}(f, BS) \) where \( BS \) is the basic set \( BS \) of a polynomial set obtained in each step in Phase 1. Methods based on this strategy can overcome the size growth difficulty for many problems that strategy (2) could not overcome. It has been used to solve 116 out of 120 problems in mechanical formula derivation [4], [5].

(4) It is impossible to list all possibly good strategies using factorization. For example, instead of (2), we might check the reducibility of basic sets produced during Phase 1. This could be a
better strategy, but we don’t have extensive experience yet.

Remark. Check of irreducibility of an asc chain generally needs factorization over extension fields, which is generally considered an expensive computation. Here we use only factorization of multivariate polynomials over the integers. We can put off the check of irreducibility over extension fields to the last step when we obtain the decomposition of the form (3.0.1) (possibly in the coarse form). For all 8 examples in the Appendix, only very few need to be factored over extension fields, and only one is reducible. This is ASC in Example (A.4).

4.2.3. Use Top Down Triangular Procedures

Phase 1 (Ritt-Wu’s Principle) is a “bottom up” triangular procedure. It has been observed that “top down” triangular procedures are much faster [1, 9]. However, top down procedures cannot insure the property (3.1.2). It was also observed by H. P. Ko in 1985 [9] that if we put the quasi asc chain obtained from top town methods into the polynomial set and work on the enlarged polynomial set, then the iteration in Phase 1 will terminate faster in many cases. In [9], a method of this type was reported. Recently, another such variant was reported in [13]. However, this approach should be combined with w-asc chain and W-prem, otherwise the size control generally cannot be insured.

4.2.4. Other Variants

As noted in [9], “There can be many variations of the above procedures and it is impossible to cover all of them. We shall list some important thoughts here.” In addition to those thoughts listed in that report [9], here we list the following observations in our experiments.

(1) A basic set of a polynomial set is not unique. We have observed that different basic sets can affect the efficiency of the algorithm in one way or other. We can further refine the partial order < of polynomials in K[y] to get a possible better control of basic sets produced in Phase 1. To refine the order < when f ~ g, we can define, e.g., f < g if lc(f) < lc(g). This was first used by Gao in [7]. Or the size of f is less than that of g; or a combination of both, etc.

(2) The number of polynomials in the sets PS in Phase 1 can grow fast. It will certainly slow down the process. However, the more polynomials in PS, the better basic set BS can be obtained. As a result, the final characteristic set could be better. There is a trade-off.

(3) Instead of (3.1.3) or (3.1.4), we can use, e.g., the following variant:

\[(3.1.4') \quad \text{Zero}(PS) = \text{Zero}(PD(ASC)) \cup \bigcup_{i=1}^{r} \text{Zero}(PS \cup ASC \cup \{lc(f_i)\}).\]

The advantage of this variant is that ASC is “almost” the characteristic set of PS \cup ASC \cup \{lc(f_i)\}. This variant speeds up the decomposition for some problems, but slows down it for other problems. It needs further refinement.

(4) The Basic sets BS produced in Phase 1 are in Ideal(PS). This is not a necessary requirement. We may require that BS be in Radical(PS) and make some polynomials in BS square free. Polynomials, such as y_2 y_3^2 etc., often occur in BS; we can replace it by y_2 y_3. In many cases, this simple replacement affects the efficiency greatly.

4.3. The Branch Control
Without any control, new polynomial sets produced in Phases 1 and 2 can be as many as tens of thousands.

4.3.1. Avoid Repeated Polynomial Sets

During our earlier experiments, we observed that there are many obvious repetitions of polynomial sets produced. For example, suppose $PS$ is the polynomial set to be decomposed. After Phase 1, we want to decompose $PS_1 = PS \cup \{I_1\}$ and $PS_2 = PS_2 \cup \{I_2\}$. In the further decomposition, we might want to decompose $PS_{1,2} = PS_1 \cup \{I_2\}$ and $PS_{2,1} = PS_2 \cup \{I_1\}$. Obviously, $PS_{1,2} = PS_{2,1}$, and we need to decompose only one of them. Such repetitions are so huge that in our earlier experiments, the algorithm could not terminate for many examples.

Example (4.3). Example (4.2) again.

Without this obvious control, the program ran more than 12 hours without terminating. Using this control, the above polynomial set was decomposed into the form of (3.0.1) with $l = 190$ and ch-number $= 11,192$. It took 17,983.4 seconds on a Symbolics 3600.

4.3.2. Take Advantages of Some Special Properties of the Polynomial Set

There are many special properties of some polynomial sets that we can use to control branching. Let us look at $PS$ in Example (A.3). Char-Set($PS$) is $ASC_1$. According to (3.1.3) we have to decompose $PS \cup \{u_5\}$, where $u_5 = lc(f_{1,7})$. However, this is redundant because $f_{1,7}$ could also be $(u_5x_1 + (u_6 - u_4)u_7 - u_1u_5)x_7 + \cdots$, thus $lc(f_{1,7}) = lc(f_{1,6})$ is the only one needed, but not $u_5$.

4.3.3. Use Zero($PS/DS$)

In decomposition of Zero($PS$), $DS$ is empty (or $DS = \{1\}$). It seems no benefits can be gained by using the trick for Zero($PS/DS$) in 3.5. It was observed in 1988 by us that we actually could use the same tricks when decomposing Zero($PS$). However, systematic experiments with this trick only started recently and to our great surprise, this trick is so effective that it reduces branches or computing time by a magnitude of 1 to 2 orders for large problems. The basic idea and related proof are almost obvious. Yet, this is one of our major techniques in control branching. We can rewrite (3.1.3):

\[
(4.3.1) \quad \text{Zero}(PS) = \text{Zero}(PD(ASC)) \cup \bigcup_{i=1}^{r} \text{Zero}(PS \cup \{lc(f_i)\}/\{lc(f_1), ..., lc(f_{i-1})\}).
\]

In this way, the final decomposition would be slightly different:

\[
(4.3.2) \quad \text{Zero}(PS) = \bigcup_{i=1}^{l} \text{Zero}(PD(ASC_i)/DS_i).
\]

Since $PS \subseteq PD(ASC_i)$ for $i = 1, ..., l$, Zero($PD(ASC_i)$) $\subseteq$ Zero($PS$). Thus, we actually can get rid of $DS_i$ in the above decomposition, and have

\[
(4.3.3) \quad \text{Zero}(PS) = \bigcup_{i=1}^{l} \text{Zero}(PD(ASC_i)).
\]
Now we can use the following techniques to control branching. During the decomposition, suppose we want to decompose Zero\( (PS' / DS) \). Let \( ASC' = \text{Char-Set}(PS') \).

1. If some \( d \in DS \) is reduced to zero by \( PS' \cup ASC' \) using some other reductions (e.g., the reduction used in the Gröbner basis method), then Zero\( (PS' / DS) \) is empty. This is one of the most important techniques to control branching in the algorithm, especially such situations happen at early stages.

2. Another technique works as follows. Let \( ASC' = f_1, ..., f_k \). If \( \text{prem}(d, ASC') = 0 \), for some \( d \in DS \), then Zero\( (PD(ASC') / DS) \) is empty. More important, we do not have to add initials of those \( f_i \) which are not used in computing \( \text{prem}(d, ASC') \) to \( PS' \).

**Example (4.4).** Using this technique, it took 167.7 sec to complete the decomposition of \( PS \) in Example (4.3) with 71 asc chains in (3.0.1) and ch-number = 145.

For this problem, the improvement is 100 times better. This is almost a universal phenomenon for most essentially large problems. See the Appendix for more information. In the Appendix we list 8 test examples. This set of 8 examples can also be served as tests for further improvement.

**4.3.4. The Dimension Theorem and the Irredundant Decomposition**

In the final decomposition in the above Example (4.4), there are 71 asc chains, thus making the result almost unmanageable. There are many redundancies. A theorem proved in [3] can reduce the redundancy greatly. According to this theorem, those asc chains whose lengths are greater than the number of polynomial set originally to be decomposed are redundant, thus can be removed from (3.0.1). The elegance of this theorem is that it is valid even for the coarse form (i.e., asc chains can be reducible). In the above example, this number is 7. Therefore, any asc chains among the 71 asc chains with lengths > 7 can be removed. Thus the number is dramatically reduced from 71 to 13.

Among those 13 asc chains, there still exist possible redundancies. To remove the redundancy completely, we need to compute the Gröbner bases of \( PD(ASC_1) \) using Chou-Schelter-Yang’s (CSY) algorithm [2]. This usually is expensive. We might use the theorems in [3] first to reduce some further redundancies. Then the number is reduced to 12. These two theorems can be stated as follows:

**Theorem (4.5).** Let \( ASC_1 \) and \( ASC_2 \) be two irreducible asc chains.

(i) \( PD(ASC_1) = PD(ASC_2) \) iff they have the same dependent variable set and \( \text{prem}(p, ASC_2) = 0 \) for all \( p \in ASC_1 \).

(ii) \( PD(ASC_1) \subseteq PD(ASC_2) \) only if \( \text{prem}(p, ASC_2) = 0 \) for all \( p \in ASC_1 \). If this is the case and \( \text{prem}(\text{lc}(p), ASC_2) \neq 0 \) for all \( p \in ASC_1 \), then \( PD(ASC_1) \subseteq PD(ASC_2) \).

**Proof.** See [3].

Theorem 4.5 can also be used to exclude the possibility that a prime is contained in another, hence to save the calculation of the Gröbner bases for some ascending chains.
Finally, we note that for an ascending chain ASC if we replace PD(ASC) by

\[ QD(ASC) = \{g \mid \exists J, Jg \in Ideal(ASC)\} \]

where \( J \) is a product of powers of the initials of the polynomials in \( ASC \), then all the results in this paper are still valid. By using \( QD(ASC) \), the algorithm to compute the Gröbner basis of \( PD(ASC) \) presented in [2] can be generalized to the following form.

**Theorem (4.6).** For an ascending chain \( ASC \) in \( K[y] \), let \( ID = Ideal(ASC, I_1 z_1 - 1, \ldots, I_p z_p - 1) \) where \( I_i \) are the initials of the polynomials in \( ASC \) and \( z_i \) are new variables. Then \( QD(ASC) = ID \cap K[y] \).

**Proof.** Let \( ASC = \{f_1, \ldots, f_p\} \). \( QD(ASC) \subseteq ID \cap K[y] \) can be proved similarly as [2]. Let \( P \in ID \cap K[y] \), then \( P = \sum B_i f_i + \sum C_i (z_i I_i - 1) \) for some polynomials \( B_i \) and \( C_i \) in \( K[y, z] \). Set \( z_i = 1/I_i \) and clear the clear the denominators. We have \( JP = \sum B_i f_i \) where \( J \) is a product of powers of the initials of the polynomials in \( ASC \), i.e., \( P \in QD(ASC) \).

Theorem 4.6 can be used to eliminate redundant components even we do not know whether the ascending chains in (3.0.1) are irreducible.

5. Conclusion

With the constant efforts by many researchers (Wu, Ko, Chou, and Gao in particular) since 1985, we have improved Ritt-Wu's algorithm at least by a magnitude of two orders. Many polynomial sets now can be easily decomposed by our program. With these improvement, we can find more applications in various areas. Furthermore, there are so many other variants which are worth experimenting. We believe that with more efforts, the algorithm can be further improved.

One of the applications is to find "weakest" non-degenerate conditions for a geometry configuration given by a set of polynomial equations and the set of parameters. The method was proposed in [6]. However, due to the inefficiency of our program, we could only use it to solve relative simple problems. With our improvement, now we can solve more complicated problems such as Feuerbach's theorem, Morley's Trisector Theorem, etc. We will discuss such application in our future work.

**References**


TR-89-21, Computer Sciences Department, The University of Texas at Austin, August 1989.


6. Appendix: Eight Test Examples

All examples below are from elementary geometry. For most problems, we rename the variables \( y_i = u_i \) for \( i = 1, \ldots, d \) and \( y_{d+i} = x_i \) for \( i = 1, \ldots, m - d \). We use ch-number to denote the number of characteristic sets produced in Phase 1. We use the method based on Theorem (4.4) to partially remove some redundancies in the decomposition. However, there are possibly further redundancies. E.g., in Example 1, \( PD(ASC_4) \subset PD(ASC_3) \). This was checked by computing the Gröbner bases of \( PD(ASC_3) \) and \( PD(ASC_4) \) using the CSY algorithm in [2]. At the end of each example, we list irredundant decomposition using the CSY algorithm.

In geometry, \( \text{Zero}(PD(ASC_1)) \) are the only non-degenerate components for all examples. Thus if we want to check whether an assertion \( g = 0 \) is valid for that configuration, then we only need to check whether \( \text{prem}(g, ASC_1) = 0 \). If it is, then the assertion \( g = 0 \) is generally true; otherwise, it is not valid in metric geometry no matter how many reasonable non-degenerate conditions are added. This is the advantage of using Formulation F1 in [3].

Example (A.1). Parallelogram \( A = (0, 0), B = (u_1, 0), C = (u_2, u_3), D = (x_2, x_1), E = (x_4, x_3) \).

It took 3.2 sec to decompose \( PS \) into 5 w-asc with ch-number = 9.

\[
\text{Zero}(PS) = \cup_{i=1}^{5} \text{Zero}(PD(ASC_i)).
\]

\[
PS =
\begin{align*}
h_1 &= u_1 x_1 - u_1 u_3 \\
h_2 &= u_3 x_2 + (-u_2 + u_1) x_1 \\
h_3 &= u_3 x_4 - u_2 x_3 \\
h_4 &= x_1 x_4 + (-x_2 + u_1) x_3 - u_1 x_1 \\

&\text{AB} \parallel CD \\
&\text{DA} \parallel CB
\end{align*}
\]

Points \( E, A \) and \( C \) are collinear. Points \( E, B \) and \( D \) are collinear.

\[
ASC_1 =
\begin{align*}
f_{1,1} &= x_1 - u_3 \\
f_{1,2} &= x_2 - u_2 + u_1 \\
f_{1,3} &= 2x_3 - u_3 \\
f_{1,4} &= 2x_4 - u_2.
\end{align*}
\]

\[
ASC_2 =
\begin{align*}
u_2 \\
u_3 \\
x_1 \\
x_2 - u_1 \\
x_1 x_4 - x_2 x_3
\end{align*}
\]

\[
ASC_3 =
\begin{align*}
u_1 \\
u_2 \\
x_3 - u_3 \\

&\text{AB} \parallel CD \\
&\text{DA} \parallel CB
\end{align*}
\]

\[
ASC_4 =
\begin{align*}
u_3 x_2 - u_2 x_1 \\
u_3 x_4 - u_2 x_3 \\
x_1 \\
x_3
\end{align*}
\]

\[
ASC_6 =
\begin{align*}
u_3 \\
x_1 \\
x_3
\end{align*}
\]

The irredundant decomposition of \( \text{Zero}(PS) \) consists of the components represented by the ascending chains \( ASC_1, ASC_2, ASC_3, ASC_4, ASC_6 \).

Example (A.2). (Simpson’s Theorem) Let \( B = (u_1, 0), A = (0, 0), C = (u_2, u_3), O = (x_2, x_1), D = (x_3, u_4), E = (x_5, x_4), F = (x_1, x_6), G = (x_3, 0) \). It took 17.8 sec to decompose \( PS \) into 8 asc chains with ch-number = 18.

\[
\text{Zero}(PS) = \cup_{i=1}^{6} \text{Zero}(PD(ASC_i)).
\]

\[
PS =
\begin{align*}
h_1 &= 2u_1 x_2 - u_1^2 \\
h_2 &= 2u_2 x_2 + 2u_3 x_1 - u_2^2 - u_2^2 \\
h_3 &= x_3^2 - 2x_2 x_3 - 2u_4 x_1 + u_4^2 \\
h_4 &= (u_2 - u_1) x_5 + u_3 x_4 + (-u_2 + u_1) x_3 - u_3 u_4 \\
h_5 &= u_3 x_5 + (-u_2 + u_1) x_4 - u_1 u_3 \\
h_6 &= u_2 x_7 + u_3 x_6 - u_2 x_3 - u_3 u_4 \\
h_7 &= u_3 x_7 - u_2 x_6
\end{align*}
\]

\[
&\text{OA} \parallel OB. \\
&\text{OA} \parallel OC. \\
&\text{OA} \parallel OD. \\
&\text{DE} \perp BC. \\
&\text{DF} \perp AC.
\]

Points \( E, B \) and \( C \) are collinear. Points \( F, A \) and \( C \) are collinear.
\[ ASC_1 = \]
\[
\begin{align*}
 f_{1,1} &= 2u_3 x_1 - u_3^2 - u_2^2 + u_1 u_2 \\
 f_{1,2} &= 2x_2 - u_1 \\
 f_{1,3} &= x_3^2 - u_1 x_3 - 2u_4 x_1 + u_3^2 \\
 f_{1,4} &= (u_3^2 + u_4^2 - 2u_2 u_4 + u_1^2)x_4 + ((-u_2 + u_1)u_3)x_3 - u_3^3 u_4 + (u_1 u_2 - u_1^2)u_3 \\
 f_{1,5} &= u_3 x_5 + (-u_2 + u_1)x_4 - u_1 u_3 \\
 f_{1,6} &= (u_3^2 + u_2^2)x_6 - u_2 u_3 x_3 - u_3^2 u_4 \\
 f_{1,7} &= u_3 x_7 - u_2 x_6
\end{align*}
\]

\[ ASC_2 = \]
\[
\begin{align*}
 u_3^3 + u_3^2 - 2u_1 u_2 + u_1^2 \\
 2u_3 x_1 - u_1 u_2 + u_1^2 \\
 2x_2 - u_1 \\
 u_3 x_3 + (-u_2 + u_1)u_4 - u_1 u_3 \\
 u_3 x_5 + (-u_2 + u_1)x_4 - u_1 u_3 \\
 (2u_1 u_2 - u_1^2)x_6 + (-u_1 u_2 + u_1^2)u_4 - u_1 u_2 u_3 \\
 u_3 x_7 - u_2 x_6
\end{align*}
\]

\[ ASC_3 = \]
\[
\begin{align*}
 u_3^2 + u_2^2 \\
 2u_3 x_1 + u_1 u_2 \\
 2x_2 - u_1 \\
 u_2 x_3 - u_2 u_4 \\
 u_3 x_5 + (-u_2 + u_1)x_4 - u_1 u_3 \\
 (2u_1 u_2 - u_1^2)x_6 + (-u_1 u_2 + u_1^2)u_4 - u_1 u_2 u_3 \\
 u_3 x_7 - u_2 x_6
\end{align*}
\]

The irredudant decomposition of Zero(PS) consists of the components represented by the ascending chains ASC_1-ASC_8.

**Example (A.3).** (Pappus’ Theorem). \( B = (u_1, 0) \), \( A = (0, 0) \), \( A_1 = (u_2, u_3) \), \( B_1 = (u_4, u_5) \), \( C = (u_6, 0) \), \( C_1 = (x_1, u_7) \), \( P = (x_3, x_2) \), \( Q = (x_5, x_4) \), \( S = (x_7, x_6) \). It took 160.6 sec to decompose PS into 12 w-asc chain with ch-number = 145.

\[ \text{Zero}(PS) = \cup_{i=1}^{12} \text{Zero}(PD(ASC_i)). \]

**Points**

- Points \( C_1, A_1 \) and \( B_1 \) are collinear.
- Points \( P, A_1 \) and \( B \) are collinear.
- Points \( P, A \) and \( B_1 \) are collinear.
- Points \( Q, A \) and \( C_1 \) are collinear.
- Points \( Q, A_1 \) and \( C \) are collinear.
- Points \( S, B_1 \) and \( C \) are collinear.
- Points \( S, B \) and \( C_1 \) are collinear.

\[ ASC_1 = \]
\[
\begin{align*}
 f_{1,1} &= (u_5 - u_3)x_1 + (-u_4 + u_2)u_7 - u_2 u_5 + u_3 u_4 \\
 h_2 &= u_3 x_3 + (-u_2 + u_1)x_2 - u_1 u_3 \\
 h_3 &= u_4 x_4 - u_4 x_2 \\
 h_4 &= u_7 x_7 - x_1 x_4 \\
 h_5 &= u_3 x_5 + (u_6 - u_2)x_4 - u_3 u_6 \\
 h_6 &= u_4 x_7 + (u_7 - u_4)x_6 - u_5 u_6 \\
 h_7 &= u_7 x_7 + (-x_1 + u_1)x_6 - u_1 u_7
\end{align*}
\]
\[
f_{1.2} = ((u_2 - u_1)u_5 - u_3u_4)x_2 + u_1u_3u_5 \\
f_{1.3} = u_3x_3 + (-u_2 + u_1)x_2 - u_1u_3 \\
f_{1.4} = (u_3x_1 + (u_5 - u_2)x_4) - u_3u_6u_7 \\
f_{1.5} = u_3x_5 + (u_6 - u_2)x_4 - u_3u_5 \\
f_{1.6} = (u_5x_1 + (u_6 - u_4)x_2 - u_1u_5)x_6 + (-u_5u_6 + u_1u_5)u_7 \\
f_{1.7} = u_5x_7 + (u_6 - u_4)x_6 - u_5u_6.
\]

\[
ASC_2 = \\
u_1 \\
u_2u_5 - u_3u_4 \\
((u_5 - u_2)u_6)u_7 \\
(u_5 - u_3)x_1 + (-u_4 + u_2)u_7 \\
u_3x_3 - u_2x_2 \\
u_3x_5 + (u_6 - u_2)x_4 - u_3u_6 \\
u_3x_7 + (u_6 - u_4)x_6 - u_3u_6
\]

\[
ASC_3 = \\
u_2 - u_1 \\
u_3 \\
u_6 - u_1 \\
u_5x_1 + (-u_4 + u_1)u_7 - u_1u_5 \\
u_3x_3 - u_4x_2 \\
u_7x_5 - x_1x_4 \\
u_5x_7 + (-u_4 + u_1)x_5 - u_1u_5
\]

\[
ASC_4 = \\
u_4 \\
u_5 \\
u_6 \\
u_3x_1 - u_2u_7 \\
u_3x_3 + (-u_2 + u_1)x_2 - u_1u_3 \\
u_3x_5 - u_2x_4 \\
u_7x_7 + (-u_1 + u_1)x_5 - u_1u_7
\]

\[
ASC_5 = \\
u_1 \\
u_2u_5 - u_3u_4 \\
u_6 \\
(u_5 - u_3)x_1 + (-u_4 + u_2)u_7 \\
u_3x_3 - u_2x_2 \\
u_3x_5 - u_2x_4 \\
u_3x_7 - u_4x_5
\]

\[
ASC_6 = \\
u_2 - u_1 \\
u_3 \\
u_7 \\
x_1 - u_1 \\
u_5x_3 - u_4x_2 \\
x_4 \\
u_5x_7 + (u_6 - u_4)x_6 - u_5u_6
\]

\[
ASC_7 = \\
u_4 \\
u_5 \\
u_7 \\
x_1 \\
u_3x_3 + (-u_2 + u_1)x_2 - u_1u_3 \\
u_3x_5 + (u_6 - u_2)x_4 - u_3u_6 \\
x_6
\]

\[
ASC_8 = \\
u_3 \\
u_4 - u_2 \\
u_5 \\
u_6 - u_2 \\
x_2 \\
x_7 - x_1x_4 \\
x_7 - (x_1 - u_1)x_6 - u_1u_7 \\
x_4
\]

\[
ASC_9 = \\
u_3 \\
u_5 \\
u_6 - u_4 \\
u_7 \\
x_1 - u_1 \\
x_2 \\
x_7 - x_1x_4 \\
x_7 - (x_1 - u_1)x_6 - u_1u_7 \\
x_4
\]

\[
ASC_{10} = \\
u_3 \\
u_5 \\
u_6 - u_2 \\
u_7 \\
x_1 \\
x_2 \\
x_4 \\
x_6
\]

\[
ASC_{11} = \\
u_2 - u_1 \\
u_3 \\
u_4 \\
u_5 \\
x_1 \\
x_2 \\
x_4 \\
x_6
\]

\[
ASC_{12} = \\
u_3 \\
u_5 \\
u_6 \\
u_7 \\
x_2 \\
x_4 \\
x_6
\]

The irredundant decomposition of \( Zero(PS) \) consists of the components represented by the ascending chains \( ASC_1 - ASC_{12} \).

**Example (A.4).** (The Butterfly Theorem) \( O = (u_1, 0), E = (0, 0), A = (u_2, u_3), B = (x_1, u_4), C = (x_3, x_2), D = (x_5, x_4), F = (0, x_5), G = (0, x_7) \).

It took 200.0 sec to decompose \( PS \) into 25 w-asc chains with ch-number = 243. If we did not use the technique in 4.3.3, it took 4046.9 sec with ch-number = 3,111.

The following decomposition took 102.7 sec with ch-number = 112.

\[
Zero(PS/DS) = \cup_{i=1}^{12} Zero(PD(ASC_i)/DS).
\]

\( DS = \{x_3 - u_3 + (x_2 - u_3), x_5 - x_1 + (x_4 - u_4)\} \). (Their geometric meanings are \( A \neq C \) and \( B \neq D \). \( PS = \))

\[
h_1 = x_1^2 - 2u_1x_1 + u_2^2 - u_3^2 - u_4^2 + 2u_1u_2 \\
h_2 = u_3x_3 - u_2x_2 \\
h_3 = x_3^2 - 2u_1x_3 + x_4^2 - u_3^2 - u_2^2 + 2u_1u_2 \\
h_4 = u_4x_4 - x_1x_4 \\
h_5 = x_5^2 - 2u_1x_5 + x_4^2 - u_3^2 - u_2^2 + 2u_1u_2 \\
h_6 = x_5^2 - u_2x_6 - u_3x_5 + u_2x_4 \\
OA \equiv OB. \quad \text{Points } C, A \text{ and } E \text{ are collinear.} \\
OA \equiv OC. \quad \text{Points } D, B \text{ and } E \text{ are collinear.} \\
OA \equiv OD. \quad \text{Points } F, A \text{ and } D \text{ are collinear.}
\]
\[ h_7 = (x_3 - x_1)x_7 - u_4x_3 + x_1x_2 \]

Points G, B and C are collinear.

\[ \text{ASC}_1 = \]
\[ f_{1,1} = x_1^2 - 2u_1x_1 + u_2^2 - u_3^2 - u_4^2 + 2u_1u_2 \]
\[ f_{1,2} = (u_3^2 + u_2^2)x_2 + u_3^3 + (u_2^3 - 2u_1u_2)u_3 \]
\[ f_{1,3} = u_4x_2 - u_2x_2 \]
\[ f_{1,4} = (2u_1x_1 + u_3^2 + u_2^2 - 2u_1u_2)x_4 + (u_3^3 + u_2^3 - 2u_1u_2)u_4 \]
\[ f_{1,5} = u_4x_5 - x_1x_4 \]
\[ f_{1,6} = (x_5 - u_2)x_6 - u_3x_5 + u_2x_4 \]
\[ f_{1.7} = (x_3 - x_1)x_7 - u_4x_3 + x_1x_2. \]

\[ \text{ASC}_2 = \]
\[ (u_3^3 + u_2^3)u_4 + u_3^3 + (u_2^3 - 2u_1u_2)u_3 \]
\[ (u_3^2 + u_2^2)x_1 + u_2u_2^3 + u_3^2 - 2u_1u_2^2 \]
\[ (u_3^2 + u_2^2)x_2 + u_3^3 + (u_2^3 - 2u_1u_2)u_3 \]
\[ u_4x_3 - u_2x_2 \]
\[ (2u_1x_1 + u_3^2 + u_2^2 - 2u_1u_2)x_4 - 2u_1u_3u_4 - u_3^3 - (u_2^3 - 2u_1u_2)u_3 \]
\[ u_4x_5 - x_1x_4 \]

\[ \text{ASC}_3 = \]
\[ u_1 \]
\[ u_3^3 + u_2^3 \]
\[ u_4x_3 - u_2x_2 \]
\[ u_3x_3 - u_2x_2 \]

\[ \text{ASC}_4 = \]
\[ u_1 \]
\[ u_3^3 + u_2^3 \]
\[ u_4x_3 - u_2x_2 \]
\[ u_3x_3 - u_2x_2 \]
\[ u_4x_5 - x_1x_4 \]
\[ (x_5 - u_2)x_6 - u_3x_5 + u_2x_4 \]

\[ \text{ASC}_5 = \]
\[ u_1 \]
\[ u_3^3 + u_2^3 \]
\[ u_4x_3 - u_2x_2 \]
\[ u_3x_3 - u_2x_2 \]
\[ u_4x_5 - x_1x_4 \]
\[ (x_5 - u_2)x_6 - u_3x_5 + u_2x_4 \]

\[ \text{ASC}_6 = \]
\[ u_1 \]
\[ u_3^3 + u_2^3 \]
\[ u_4x_3 - u_2x_2 \]
\[ u_3x_3 - u_2x_2 \]
\[ u_4x_5 - x_1x_4 \]
\[ (x_5 - u_2)x_6 - u_3x_5 + u_2x_4 \]

The irredundant decomposition of \textit{Zero}(PS/DS) consists of the components represented by the ascending chains \text{ASC}_1, \text{ASC}_2, \text{ASC}_6, \text{ASC}_7, and \text{ASC}_{12}.

Example (A.5). (the 9-Point Theorem) \( B = (u_1,0), A = (0,0), C = (u_2,u_3), D = (x_2,x_1), E = (x_4,x_3), F = (u_2,0), M = (x_5,0), N = (x_7,x_6). \)

It took 70.8 sec to decompose PS into 17 w-asc chains with ch-number \( = 51 \). If we did not use the technique in
The following decomposition took 21.8 sec and the ch-number is 13.

\[
\text{Zero}(PS/DS) = \bigcup_{i=1}^{\delta} \text{Zero}(PD(ASC_i))/DS.
\]

\[
DS = \{u_1, u_3\} \quad \text{(Geometric meaning: A, B, C are not collinear). PS =}
\]

\[
\begin{align*}
    h_1 &= u_3 x_2 + (-u_2 + u_1) x_1 - u_1 u_3 \\
    h_2 &= (u_2 - u_1) x_2 + u_3 x_1 \\
    h_3 &= u_3 x_4 - u_2 x_3 \\
    h_4 &= u_2 x_4 + u_3 x_3 - u_1 u_2 \\
    h_5 &= 2 x_5 - u_1 \\
    h_6 &= (2 x_4 - 2 u_2) x_7 + 2 x_3 x_6 - x_2^2 - x_3^2 + u_2^2 \\
    h_7 &= (2 x_2 - 2 u_2) x_7 + 2 x_1 x_6 - x_2^2 - x_1^2 + u_2^2
\end{align*}
\]

\[
ASC_1 =
\]

\[
\begin{align*}
    f_{1,1} &= (u_2^3 + u_2^2 - 2 u_1 u_2 + u_1^2) x_1 + (u_1 u_2 - u_1^2) u_3 \\
    f_{1,2} &= ((u_2 - u_1) x_2 + u_3 x_1 \\
    f_{1,3} &= (u_2^2 + u_2^3) x_3 - u_1 u_2 u_3 \\
    f_{1,4} &= u_2 x_4 + u_3 x_3 - u_1 u_2 \\
    f_{1,5} &= 2 x_5 - u_1 \\
    f_{1,6} &= (2 x_1 x_4 + (-2 x_2 + 2 u_2) x_3 - 2 u_2 x_1) x_6 + (x_2 - u_2) x_4^2 + (-x_2^2 - x_1^2 + u_2^2) x_4 + (x_2 - u_2) x_3^2 + u_2 x_2^2 - u_2^2 x_2 + u_2 x_1^2 \\
    f_{1,7} &= (2 x_2 - 2 u_2) x_7 + 2 x_1 x_6 - x_2^2 - x_1^2 + u_2^2
\end{align*}
\]

\[
ASC_2 =
\]

\[
\begin{align*}
    u_2 - u_1 \\
    x_1 \\
    x_2 - u_1 \\
    (u_2^2 + u_1^2) x_3 - u_2^2 u_3 \\
    u_3 x_4 - u_1 x_3 \\
    2 x_5 - u_1 \\
    (2 x_4 - 2 u_1) x_7 + 2 x_3 x_6 - x_2^2 - x_3^2 + u_1^2
\end{align*}
\]

\[
ASC_3 =
\]

\[
\begin{align*}
    u_2 u_3 \\
    (u_2^3 + u_1^2) x_1 - u_2^2 u_3 \\
    x_1 x_2 - u_3 x_1 \\
    u_3 x_4 \\
    x_4 \\
    2 x_5 - u_1 \\
    2 x_2 x_7 + 2 x_1 x_6 - x_2^2 - x_1^2 + u_2^2
\end{align*}
\]

\[
ASC_4 =
\]

\[
\begin{align*}
    u_3^2 + u_2^2 - u_1 u_2 \\
    x_1 - u_3 \\
    x_2 - u_2 \\
    x_3 - u_3 \\
    x_4 - u_2 \\
    2 x_5 - u_1 \\
    2 u_3 x_6 + u_3^2 - u_1 u_2
\end{align*}
\]

The irredundant decomposition of \(\text{Zero}(PS/DS)\) consists of the components represented by the ascending chains \(ASC_1\)–\(ASC_4\).

**Example (A.6).** (Feuerbach’s Theorem) \(A = (u_1, 0), D = (0, 0), I = (0, u_2), B = (u_3, 0), C = (x_2, x_1), M_1 = (x_3, 0), M_2 = (x_4, x_3), M_3 = (x_6, x_7), N = (x_9, x_8).\)

It took 45.8 sec to decompose \(PS\) into 8 w-asc chains with ch-number = 31.

\[
\text{Zero}(PS) = \bigcup_{i=1}^{\delta} \text{Zero}(PD(ASC_i)).
\]

\[
\begin{align*}
    h_1 &= (2 u_2 u_3^2 - 2 u_1 u_2 u_3) x_2 + (u_2^3 - u_1 u_2^2 - u_2^2 u_3 + u_1 u_3^2) x_1 - 2 u_2 u_3^2 + 2 u_1 u_2 u_3^2 \\
    h_2 &= (2 u_1 u_2 u_3 - 2 u_1^2 u_2) x_2 + ((-u_2^2 + u_1^2) u_3 + u_1 u_2^2 - u_3^2) x_1 - 2 u_2 u_3^2 + 2 u_1 u_2 u_3^2 \\
    h_3 &= 2 x_3 - u_3 - u_1 \\
    h_4 &= 2 x_4 - x_2 - u_1 \\
    h_5 &= 2 x_5 - x_1 \\
    h_6 &= 2 x_6 - x_2 - u_3 \\
    h_7 &= 2 x_7 - x_1 \\
    h_8 &= (2 x_6 - 2 x_3) x_9 + 2 x_7 x_8 - x_2^2 - x_3^2 + u_3^2 \\
    h_9 &= (2 x_4 - 2 x_3) x_9 + 2 x_5 x_8 - x_2^2 - x_4^2 + u_4^2
\end{align*}
\]

\[
ASC_1 =
\]

16
The redundant decomposition of \( \text{Zero}(PS) \) consists of the components represented by the ascending chains \( \text{ASC}_1 - \text{ASC}_4, \text{ASC}_6, \) and \( \text{ASC}_7. \)

**Example (A.7).** (Morley’s Trisector Theorem) \( B = (u_1, 0), A = (0, 0), D = (u_2, u_3), C = (x_2, x_1), X = (x_3, 0), F = (x_5, x_4), E = (x_7, x_6). \) It took 947.5 seconds to decompose \( \text{Zero}(PS) \) into 33 w-asc chains with ch-number = 106.

The following decomposition took 663.8 sec with ch-number = 36.

\[
\text{Zero}(PS/DS) = \bigcup_{i=1}^{6} \text{Zero}(PD(\text{ASC}_i)/DS).
\]

**DS = \{ u_1, x_1 \} (A, B, and C are not collinear.) PS =**

\[
h_1 = (u_3^3 + (-3u_2^2 + 6u_1 u_2 - 3u_1^2) u_3) x_2 + ((-3u_2 + 3u_1) u_3^3 + u_3^2 - 3u_1 u_2^2 + 3u_1^2 u_2 - u_1^3) x_1 - u_1 u_2^3 + (3u_1 u_2^2 - 6u_1^2 u_2 + 3u_1^3) u_3 
\]

\[
\text{tan}(\angle CBA) - \text{tan}(\angle DBA) = 0.
\]

\[
h_2 = (u_3^2 - 3u_2^2 u_3) x_2 + (-3u_2 u_3^2 + u_3^3) x_1 
\]

\[
\text{tan}(\angle CAB) - \text{tan}(\angle DAB) = 0.
\]

\[
h_3 = \pm \sqrt{3}. 
\]

\[
h_4 = (u_1 u_3 x_2 - u_1 u_2 x_1) x_5 + (u_1 u_3 x_2 + u_1 u_2 x_1) x_4 
\]

\[
\text{tan}(\angle DAB) = \text{tan}(\angle CAF).
\]

\[
h_5 = ((u_2^2 - u_1^2) x_3 - u_1 u_2 x_1) x_3 + u_1 u_2 x_2 + (u_2^3 + u_2^2 - u_1 u_2) x_1) x_5 + ((u_1 u_2 x_2 + (u_2^3 + u_2^2 - u_1 u_2) x_1) x_3 + (-u_2^2 - u_2^3 + u_1 u_2) x_2 + u_1 u_2 x_1) x_4 + ((-u_2^2 - u_2^3 + u_1 u_2) x_2 + (-u_2^3 - u_2^2 + u_1 u_2) x_1) x_5 - u_1 u_2 x_3 - u_1 u_2 x_1) x_5 
\]

\[
\text{tan}(\angle BAD + \angle DBA + \angle ACB) = \pm \sqrt{3}.
\]

\[
h_6 = (u_1 u_3 x_2 + (-u_1 u_2 + u_1^2) x_1 - u_2^3 x_3) x_7 + ((u_1 u_2 - u_1^2) x_3 + u_1 u_3 x_1 - u_1^2 u_2 + u_1^3) x_5 - u_1 u_2 x_3 + (u_1 u_2 - u_1^3) x_1 + u_1^3 x_3
\]

17
\[ h_7 = ((2x_1x_2 - u_1x_1)x_5 + (-x_2^2 + u_1x_2 + x_1^2)x_4 - x_1x_2^2 - x_2^3)x_7 + ((-x_2^2 + u_1x_2 + x_1^2)x_5 + (-2x_1x_2 + u_1x_1)x_4 + x_2^2 - u_1x_2^2 + x_1^2)x_6 + (-x_1x_2^2 - x_2^3)x_5 + (x_2^2 - u_1x_2^2 + x_1^2)x_4 + u_1x_1x_2^3 + u_1x_1^3 \tan(ACF) = \tan(ECB). \]

\[ \tan(ABD) = \tan(EBC). \]

\[ ASC_1 = \]
\[ f_{1.1} = (3u_3^2 + (6u_2^2 - 6u_1u_2 - u_1^2)u_2^2 + 3u_2^2 - 6u_1u_2^2 + 3u_1^2u_2)u_1^2 + u_3^2 + (6u_2^2 - 6u_1u_2 + 3u_2^2)u_3^2 + (-9u_2^2 + 18u_1u_2^2 - 9u_2^2)u_3^2) \]
\[ f_{1.2} = (u_2^2 - 3u_2^2)u_3^2 + (-3u_2^3 + u_3^2)x_1 \]
\[ f_{1.3} = x_3^2 - 3 \]
\[ f_{1.4} = ((u_2 - u_1)u_3^2 + u_3^2 - u_1u_2^2)x_3 + u_3^2 + u_3^2)x_4 + ((u_1^2 + (u_2 - u_1)u_2)x_2 + (-u_2^2 - u_3 + u_1u_2^2)x_1)x_3 + u_1u_2^2x_2 - u_1u_2x_1 \]
\[ f_{1.5} = (u_2x_2 - u_2x_1)x_5 + (u_2x_2 - u_3x_1)x_4 \]
\[ f_{1.6} = ((u_3x_2 + (u_2 - u_1)x_1)x_5 + ((u_2 - u_1)x_2 + u_3x_1)x_4 - u_3x_2^2 - u_3x_1^2)x_6 + (u_2x_1x_2 + (-u_2 + u_1)x_2 - u_1u_3x_2)x_5 + u_3x_2^2 + ((u_2 - u_1)x_1 + u_1u_3)x_2)x_4 \]
\[ f_{1.7} = (u_3x_2 + (-u_2 + u_1)x_1 - u_1u_3)x_7 + ((u_3x_1)x_2 + u_3x_1 - u_1u_3 + u_3^2)x_6 - u_1u_3x_2 + (u_1u_2 - u_2^2)x_1 + u_1^2u_3. \]

\[ ASC_2 = \]
\[ f_{2.1} = u_3^2 + u_2^2 - 2u_1u_2 + u_1^2 \]
\[ f_{2.2} = u_1^2u_3x_1 + 4u_2^2 - 10u_1u_2^2 + 9u_2^2u_2 - 4u_3^2u_2 + u_1^4 \]
\[ f_{2.3} = u_3x_2 + (-u_2 + u_1)x_1 - u_1u_3 \]
\[ f_{2.4} = x_1^3 - 3 \]
\[ f_{2.5} = ((u_2^2 + 3u_1u_2 + u_2^2)x_3 + (2u_2 - u_1)u_2^2)x_4 + ((u_2 - u_1)u_3)x_2 + (-u_2^2 + u_1u_2)x_1)x_3 + (-u_2^2 + 2u_1u_2 - u_2^2)x_2 - u_2u_3x_4 \]
\[ f_{2.6} = (u_2x_2 - u_2x_1)x_5 + (u_2x_2 - u_3x_1)x_4. \]

\[ ASC_3 = \]
\[ f_{3.1} = u_3^2 + u_2^2 \]
\[ f_{3.2} = u_1^2u_3x_1 + 4u_2^2 - 6u_1u_2^2 + 3u_2^2u_2 \]
\[ f_{3.3} = u_1^2 - 4u_2^2 + 6u_1u_2^2 - 3u_2^2u_2 \]
\[ f_{3.4} = x_3^2 - 3 \]
\[ f_{3.5} = (u_1u_3x_5 - u_1u_3x_4)x_6 + (2u_2^2 - 2u_1u_2^2)x_5 - (2u_2^2 - 2u_1u_2^2)u_3x_4 \]
\[ f_{3.6} = u_1x_7 + u_2^2 - 2u_1u_2 \]

\[ ASC_4 = \]
\[ f_{4.1} = u_3^2 + u_2^2 \]
\[ f_{4.2} = u_1^2u_3x_1 + 4u_2^2 - 6u_1u_2^2 + 3u_2^2u_2 \]
\[ f_{4.3} = u_1^2 - 4u_2^2 + 6u_1u_2^2 - 3u_2^2u_2 \]
\[ f_{4.4} = x_3^2 - 3 \]
\[ f_{4.5} = u_3x_5 - u_2x_4 \]
\[ f_{4.6} = (u_2 - u_1)u_3x_7 - (2u_2^2 - 2u_1u_2 + u_2^2)x_6 - (2u_1u_2 - u_2^2)u_3x_4 \]

\[ ASC_5 = \]
\[ f_{5.1} = 2u_2 - u_1 \]
\[ f_{5.2} = 4u_2^2 + u_1^2 \]
\[ f_{5.3} = 4u_1x_1 + u_1^2 \]
\[ f_{5.4} = 2u_3x_2 + u_1x_1 \]
\[ f_{5.5} = x_3^2 - 3. \]

\[ ASC_6 = \]
\[ f_{6.1} = u_2 - u_1 \]
\[ f_{6.2} = u_2^2 + u_1^2 \]
\[ f_{6.3} = u_3x_1 + u_1^2 \]

18
\[
f_{6.4} = x_2 - u_1 \\
f_{6.5} = x_2^2 - 3 \\
f_{6.6} = u_3 x_5 - u_1 x_4 \\
f_{6.7} = x_6.
\]

The irredundant decomposition of \( \text{Zero}(PS/DS) \) consists of the components represented by the ascending chains \( \text{ASC}_{1-3} \).

**Example (A.8).** (The \( 8_3 \) Configuration) \( A = (0, 0), B = (y_1, 0), D = (y_2, 0), C = (y_3, y_8), E = (y_4, y_9), F = (y_5, y_{10}), G = (y_6, y_{11}), H = (y_7, y_{12}). \)

It took 1344.8 sec to decompose \( PS \) into 15 w-asc chains with ch-number = 426. If we did not use the technique in 4.3.3, it ran for more than 24 hours without terminating.

\[
\text{Zero}(PS) = \bigcup_{i=1}^{15} \text{Zero}(PD(\text{ASC}_i)).
\]

\[
\begin{align*}
PS &= \sum_{i=1}^{15} \text{Zero}(PD(\text{ASC}_i)) \\
&= h_1 = y_3 y_{12} - y_7 y_8 \\
h_2 &= y_3 y_{11} - y_6 y_{10} \\
h_3 &= (y_3 - y_1) y_9 + (-y_4 + y_1) y_8 \\
h_4 &= (y_6 - y_1) y_{12} + (-y_7 + y_1) y_{11} \\
h_5 &= (y_3 - y_2) y_{10} + (-y_5 + y_2) y_8 \\
h_6 &= (y_4 - y_2) y_{11} + (-y_6 + y_2) y_9 \\
h_7 &= (y_5 - y_4) y_{12} + (-y_7 + y_4) y_{10} + (y_7 - y_5) y_9.
\end{align*}
\]

\[
\begin{align*}
\text{ASC}_1 &= \\
f_{1.1} &= ((y_2^2 - y_1 y_2 + u_1^2) y_3 + ((y_1 y_2 - 2 y_1^2) y_3 - 2 y_1 y_2^2 + y_1^2 y_2) y_4 + (y_1^2 y_3^2 - y_1 y_2 y_3 + u_1 y_2^2) y_5 + ((-2 y_2 + y_1 y_2) y_3 + y_1 y_2^2 - 2 y_1^2 y_2) y_6) y_7 + (y_1^2 y_3^2 - y_1 y_2 y_3 + u_1 y_2^2) y_8 + u_1^2 y_2 y_3) y_9 \\
f_{1.2} &= ((y_2 y_3 - y_1 y_3 + y_4 + (-y_2 + y_1) y_3 + (-y_2 y_3 + y_1 y_2) y_4 + y_2^2 y_3 + y_1 y_2^2) y_5 + (-y_2 y_3 - y_2^2) y_6) y_7 + ((-2 y_2 + y_1 y_2) y_3 - y_1 y_2^2 + y_1 y_2 y_3) y_8 + (y_2 y_3 - y_1 y_2 y_3 + u_1 y_2^2) y_9 \\
f_{1.3} &= ((y_2 y_3 - y_1 y_3 + y_4 + (-y_2 y_3 + y_1 y_2) y_4 + y_2^2 y_3 - y_1 y_2^2) y_5 + (-y_2 y_3 - y_2^2) y_6) y_7 + ((-2 y_2 + y_1 y_2) y_3 - y_1 y_2^2 + y_1 y_2 y_3) y_8 + (y_2 y_3 - y_1 y_2 y_3 + u_1 y_2^2) y_9 \\
f_{1.4} &= (y_3 - y_1) y_9 + (-y_4 + y_1) y_8 \\
f_{1.5} &= (y_3 - y_2) y_{10} + (-y_5 + y_2) y_8 \\
f_{1.6} &= (y_4 - y_2) y_{11} + (-y_6 + y_2) y_9 \\
f_{1.7} &= (y_3 y_{12} - y_7 y_8).
\end{align*}
\]

\[
\begin{align*}
\text{ASC}_2 &= \sum_{i=1}^{15} \text{Zero}(PD(\text{ASC}_i)) \\
y_1 &= y_1 \\
y_2 &= y_2 \\
y_3 &= y_3 \\
y_6 &= y_6 \\
y_6 &= y_4 y_{10} - y_5 y_9 \\
y_1 &= y_4 y_{11} - y_6 y_9 \\
(y_5 - y_4) y_{12} - (y_7 - y_4) y_9 + (y_7 - y_5) y_9 & (y_5 - y_4) y_{12} - y_7 y_{10} + y_7 y_9
\end{align*}
\]

\[
\begin{align*}
\text{ASC}_3 &= \sum_{i=1}^{15} \text{Zero}(PD(\text{ASC}_i)) \\
y_3 &= y_4 - y_1 \\
y_5 &= -y_5 + y_2 \\
y_6 &= y_7 \\
y_8 &= y_9 \\
y_9 &= y_10 \\
y_{10} &= y_{11} \\
y_{11} &= y_{12}
\end{align*}
\]

\[
\begin{align*}
\text{ASC}_4 &= \sum_{i=1}^{15} \text{Zero}(PD(\text{ASC}_i)) \\
y_2 &= y_2 \\
y_5 &= y_6 \\
y_7 &= y_7 \\
y_6 &= y_5 y_{10} + (-y_4 + y_1) y_6 \\
y_5 &= y_4 - y_1) y_{10} + (-y_5 + y_1) y_8 \\
y_4 &= (y_4 - y_1) y_{11} + (-y_6 + y_1) y_9 \\
y_{12} &= y_6 - y_6
\end{align*}
\]

\[
\begin{align*}
\text{ASC}_5 &= \sum_{i=1}^{15} \text{Zero}(PD(\text{ASC}_i)) \\
y_3 &= y_4 - y_1 \\
y_5 &= -y_5 + y_2 \\
y_6 &= y_7 \\
y_8 &= y_9 \\
y_9 &= y_{10} \\
y_{10} &= y_{11} \\
y_{11} &= y_{12}
\end{align*}
\]

\[
\begin{align*}
\text{ASC}_6 &= \sum_{i=1}^{15} \text{Zero}(PD(\text{ASC}_i)) \\
y_3 &= y_4 - y_1 \\
y_5 &= y_6 - y_2 \\
y_6 &= y_7 - y_1 \\
y_7 &= y_4 - y_4 \\
y_8 &= y_9 \\
y_{10} &= y_{11} \\
y_{11} &= y_{12}
\end{align*}
\]

\[
\begin{align*}
\text{ASC}_7 &= \sum_{i=1}^{15} \text{Zero}(PD(\text{ASC}_i)) \\
y_3 &= y_3 - y_1 \\
y_5 &= y_6 - y_2 \\
y_6 &= y_7 - y_4 \\
y_8 &= y_9 \\
y_9 &= y_{10} \\
y_{10} &= y_{11} \\
y_{11} &= y_{12}
\end{align*}
\]

\[
\begin{align*}
\text{ASC}_8 &= \sum_{i=1}^{15} \text{Zero}(PD(\text{ASC}_i)) \\
y_3 &= y_3 - y_1 \\
y_5 &= y_6 - y_2 \\
y_6 &= y_7 - y_4 \\
y_8 &= y_9 \\
y_9 &= y_{10} \\
y_{10} &= y_{11} \\
y_{11} &= y_{12}
\end{align*}
\]

\[
\begin{align*}
\text{ASC}_9 &= \sum_{i=1}^{15} \text{Zero}(PD(\text{ASC}_i)) \\
y_3 &= y_3 - y_1 \\
y_5 &= y_6 - y_2 \\
y_6 &= y_7 - y_4 \\
y_8 &= y_9 \\
y_9 &= y_{10} \\
y_{10} &= y_{11} \\
y_{11} &= y_{12}
\end{align*}
\]

\[
\begin{align*}
\text{ASC}_{10} &= \sum_{i=1}^{15} \text{Zero}(PD(\text{ASC}_i)) \\
y_3 &= y_3 - y_1 \\
y_5 &= y_6 - y_2 \\
y_6 &= y_7 - y_4 \\
y_8 &= y_9 \\
y_9 &= y_{10} \\
y_{10} &= y_{11} \\
y_{11} &= y_{12}
\end{align*}
\]
<table>
<thead>
<tr>
<th>$ASC_{11}$ =</th>
<th>$ASC_{12}$ =</th>
<th>$ASC_{13}$ =</th>
<th>$ASC_{14}$ =</th>
<th>$ASC_{15}$ =</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_1$</td>
<td>$y_1$</td>
<td>$y_2$</td>
<td>$y_1$</td>
<td>$y_8$</td>
</tr>
<tr>
<td>$y_2$</td>
<td>$y_5 - y_3$</td>
<td>$y_4 - y_3$</td>
<td>$y_2$</td>
<td>$y_9$</td>
</tr>
<tr>
<td>$y_3y_9 - y_4y_8$</td>
<td>$-y_6 + y_4$</td>
<td>$y_7 - y_6$</td>
<td>$y_3$</td>
<td>$y_{10}$</td>
</tr>
<tr>
<td>$y_3y_10 - y_5y_8$</td>
<td>$y_3y_5 - y_4y_8$</td>
<td>$y_6 - y_8$</td>
<td>$y_4$</td>
<td>$y_{11}$</td>
</tr>
<tr>
<td>$y_4y_{11} - y_6y_9$</td>
<td>$y_{10} - y_6$</td>
<td>$y_3y_{10} - y_5y_8$</td>
<td>$y_5$</td>
<td>$y_{12}$</td>
</tr>
<tr>
<td>$y_6y_8 - y_7y_8$</td>
<td>$y_{11} - y_9$</td>
<td>$y_3y_{11} - y_6y_8$</td>
<td>$y_6$</td>
<td>$y_{12}$</td>
</tr>
<tr>
<td></td>
<td>$y_3y_{12} - y_7y_8$</td>
<td>$y_3y_{12} - y_6y_8$</td>
<td>$y_7$</td>
<td></td>
</tr>
</tbody>
</table>

The irredundant decomposition of Zero($PS$) consists of the components represented by the ascending chains $ASC_1$, $ASC_2$, $ASC_4-ASC_0$, $ASC_{11}-ASC_{13}$, and $ASC_{15}$. 