Applications of Bar Code to Involutive Divisions and a “Greedy” Algorithm for Complete Sets

Michela Ceria

Received: 29 September 2021 / Revised: 5 October 2022 / Accepted: 13 November 2022 / Published online: 29 November 2022
© The Author(s), under exclusive licence to Springer Nature Switzerland AG 2022

Abstract Given a finite set of terms \( U \) in \( n \) variables, we describe an algorithm which finds – if it exists – an ordering on the variables such that \( U \) is a complete set according to Janet involutive division. The algorithm, based on Bar Codes for monomial ideals, is able to adjust the variables ordering with a sort of backtracking technique, thus allowing to find the desired ordering without trying all the \( n! \) possible ones.

Keywords Janet division · Bar Codes · Completeness

1 Introduction

Involutive divisions and involutive bases are a very important topic in Computer Algebra. Their theory dates back to the works by Janet [34–37]. Given the polynomial ring \( P := k[x_1, \ldots, x_n] \) in the \( n \) variables \( x_1, \ldots, x_n \) and coefficients in the field \( k \), the semigroup of terms \( T \subset P \), is given by \( T := \{ x_1^{\gamma_1} \cdots x_n^{\gamma_n} | \gamma_1, \ldots, \gamma_n \in \mathbb{N} \} \).

In [34], Janet takes a semigroup/monomial ideal \( J \subset T \) and its minimal generating set (that we will denote by \( G(J) \)). He associates to each generator a subset of variables, that are called multiplicative. Moreover, he decomposes the semigroup ideal \( J \) in disjoint subsets called cones and describes a procedure (called completion) to construct this decomposition. For each term \( v \in T \), there is a unique way to write \( v = tu \), with \( t \in G(J) \) and \( u \) a product of powers of \( t \)'s multiplicative variables. In this context, when we have to reduce a term \( w \) modulo an ideal which has \( J \) as initial ideal, the polynomial to use is the only one whose leading term generates the cone containing \( w \).

In [34], Janet aims to describe Riquier’s formulation [42] of the description for the general solutions of a PDE problem, and for this aim he gives also an analogous decomposition for the escalier associated to \( J \), namely \( N(J) := T \setminus J \). In his following works [35–37], he gives a new decomposition, named involutive, which is behind both Gerdt-Blinkov [21–23] procedure to compute Groebner bases and Seiler’s involutivity theory [45]. His first aim is to give an interpretation by means of multiplicative variables of Cartan’s solution to PDE problems [1–3] (whence the name involutive). The second aim is to evaluate in his theory’s framework the notion of generic initial ideal introduced by Delassus [15–17] and the correction of his mistake by Robinson [43,44] and Gunther [29,30], in memory of Vladimir Gerdt.
who remarks that the notion requires \( J \) to be Borel-fixed (an equivalent modern reformulation has been proposed by Galligo \[20\], who merges Hironaka and Grauert’s ideas \[27,33\]; see also \[18,28\]).

In \[36\] Janet presents, as *nouvelle formes canoniques*, Delassus, Robinson and Gunther’s results. Moreover, he gives a comparison with the canonical forms one can deduce from an involutive basis. In \[37\, p.62\], given a homogeneous ideal \( I \) of \( \mathbb{P} \) in generic coordinates, he restates Riquier’s completion in terms of a Macaulay-like construction, iteratively computing the vector spaces \( I_d := \{ f \in I : \deg(f) = d \} \) until a precisely stated formula, called *Cartan test*, grants that Castelnuovo-Mumford regularity \[39, pg.99\] has been reached. Thus Castelnuovo-Mumford regularity was obtained for the first time by Janet via this explicit algorithm. This would allow him to consider the semigroup ideal \( T(I) \) of the leading terms with respect to deglex (in the sense of Groebner basis theory) and get the *involutive reduction* required by Riquier’s procedure. The formal definition of involutive division is due to Gerdt-Blinkov \[21,22\].

Bar Codes, introduced in \[5,6\], are a compact, bidimensional representation for finite sets of terms \( M \subset T \) in any number of variables. In particular, if the set \( M = N(I) \) is the Groebner escalier of a zerodimensional ideal \( I \) of \( \mathbb{P} \) with respect to Lex, many of the ideal’s properties can be directly read from its Bar Code. As an example, in \[9\], Bar Codes are the main tool to develop a combinatorial algorithm which, given a finite set of simple points, computes the lexicographical Groebner escalier of its vanishing ideal. This algorithm is an alternative to those by Cerlienco-Mureddu \[12–14\] and by Felszeghy-Ráth-Rónyay \[19\], which keeps the former algorithm’s iterativity, though reaching a complexity which is near to that of the latter one. In \[5\], we use Bar Codes to define and prove that there is a 1 \(-\) 1 correspondence between zerodimensional (strongly) stable ideals in two or three variables and some partitions of their affine Hilbert polynomial, which is constant. Now, we are focusing on the properties of Bar Codes connected to involutive divisions, for which Bar Codes have already proved to be a good technology \[7,8\]. For a general overview of Bar Codes’ applications see \[6\].

In this paper, we discuss how the Bar Code associated to a finite set of terms (which is non-necessarily an order ideal) allows to decide whether that set is complete according to Janet’s definition \[34\]. Moreover, we give an algorithm to check whether there is a variables ordering such that a given set of terms is complete. We remark that the aim of this paper is not to build a completion, as it was for \[22,45\], but to check whether there is need of a completion for every variables ordering.

We want to remark that such a topic has some connections to the study of Stanley decompositions and Stanley depth. Indeed, Janet decomposition for a complete set is a Stanley decomposition which is read from the complete set in an easy way. Anyway, as stated by Herzog \[32\],

*Janet decompositions from the viewpoint of Stanley depth are not optimal. They rarely give Stanley decompositions providing the Stanley depth of a monomial ideal. However one obtains the result that the Stanley depth of a monomial ideal is at least 1.*

and, actually, this paper places itself in the field of study mainly developed by Gerdt-Blinkov \[21–23\] and Seiler \[45\], which has aims and language that are different from those of Stanley depth.

After Sect. 2, devoted to notation (see Sect. 2.1) and to a recap on Bar Codes (Sect. 2.2), we describe Janet decomposition into multiplicative/non-multiplicative variables (Sect. 3), recalling how to use the Bar Code to get it from a finite set of terms. Moreover, we deal with complete sets, explaining how also completeness can be read from a suitable Bar Code. In Sect. 4, then, we explain an algorithm to detect a variable ordering (if it exists) such that a given set of terms is complete according to that ordering. The algorithm constructs a Bar Code from the maximal to the minimal variable, adjusting the variables ordering with a sort of backtracking technique, and allowing to construct the desired ordering without trying all the \( n! \) possible orderings.
2 Notation and Preliminaries

2.1 Some General Notation

The principal reference for the notation used in this paper are the volumes of [38].

We start considering a field \( k \) of any characteristic, and defining over it the polynomial ring \( P := k[x_1, \ldots, x_n] \) in the \( n \) variables \( x_1, \ldots, x_n \); we also consider the semigroup of terms in the same variables \( T := \{x_1^{\gamma_1} \cdots x_n^{\gamma_n} | \gamma_1, \ldots, \gamma_n \in \mathbb{N}\} \). If \( A \subseteq \{1, \ldots, n\} \) then \( T[A] := \{x_1^{\gamma_1} \cdots x_n^{\gamma_n} \in T | \gamma_i \neq 0 \Rightarrow i \in A\} \).

The degree of a term \( t = x_1^{\gamma_1} \cdots x_n^{\gamma_n} \) is defined as \( \deg(t) = \sum_{i=1}^{n} \gamma_i \), while its \( h \)-degree, for \( h \in \{1, \ldots, n\} \), is \( \deg_T(t) := \gamma_h \).

We can define on \( T \) a semigroup ordering, namely a total ordering \(<\) such that it holds \( t_1 < t_2 \Rightarrow st_1 < st_2 \), for each \( s, t_1, t_2 \in T \).

A semigroup ordering which has also the property to be a well ordering is called term ordering; the only term ordering we consider in this paper is the lexicographical ordering (also called Lex) with \(^1 \) \( x_1 < \ldots < x_n; x_1^{\delta_1} \cdots x_n^{\delta_n} < x_1^{\gamma_1} \cdots x_n^{\gamma_n} \) if and only if there is \( j \) such that \( \gamma_j < \delta_j, \gamma_i = \delta_i \), for each \( i > j \). Given \( t \in T \), we call \( \max(t) \) (resp. \( \min(t) \)) the maximal (resp. minimal) variable dividing \( t \). Once fixed a semigroup/term ordering \(<\) on \( T \), for each \( f \in P \) we define its leading term \( T(f) \) to be its maximal term with respect to \(<\).

We call semigroup ideal a subset \( J \subseteq T \) such that, if a term \( t \) is in \( J \), then the product \( st \) is in \( J \) as well, for each \( s \in T \); an order ideal is instead a subset \( N \subseteq T \) such that, if \( t \in N \), then \( s \in N \), for each \( s \) that is a divisor of \( t \). It is quite straightforward to show that \( N \subseteq T \) is an order ideal if and only if \( T \setminus N = J \) is a semigroup ideal.

The minimal generating set of a semigroup ideal \( J \subseteq T \) is called monomial basis and denoted by \( G(J) \). We associate to \( J \) also the order ideal \( N(J) := T \setminus J \).

Now, considered a set of polynomials \( G \subseteq P \), we define \( T[G] := \{T(g), g \in G\} \) and \( T(G) := \{tT(g), t \in T, g \in G\} \); the latter is the semigroup ideal of leading terms. If \( I \) is an ideal of \( P \), the monomial basis of \( T(I) = T[\{I\}] \) is named monomial basis of \( I \). The usual notation is again \( G(I) \). We call instead Groebner escalier of \( I \) the order ideal \( N(I) := T \setminus T(I) \).

2.2 Bar Code for Monomial Ideals: a Light Recap

In this section we give a brief summary of the definitions and properties concerning Bar Codes that we will need in the rest of the paper. For more details, see [5,6].

**Definition 2.1** A Bar Code \( B \) is a diagram composed by segments, (the bars), superimposed in horizontal rows, satisfying the Condition \( a \), below. Denote by \( B_j^{(i)} \) the \( j \)-th bar (from left to right) of the \( i \)-th row (from top to bottom), \( 1 \leq i \leq n \), i.e. the \( j \)-th \( i \)-bar and by \( \mu(i) \) the number of bars of the \( i \)-th row:

a. \( \forall i, j, 1 \leq i \leq n - 1, 1 \leq j \leq \mu(i), \exists! \exists! j \in \{1, \ldots, \mu(i) + 1\} \) s.t. \( B_j^{(i)} \) lies under \( B_j^{(i)} \).

The notation \( l_1(B_j^{(1)}) := 1 \), for each \( j \in \{1, 2, \ldots, \mu(1)\} \), indicates the \( 1 \)-length of the 1-bars. This is also called simply length. The notation \( l_i(B_j^{(k)}) \), \( 2 \leq k \leq n, 1 \leq i \leq k - 1, 1 \leq j \leq \mu(k) \), instead, is the \( i \)-length of \( B_j^{(k)} \), namely the number of \( i \)-bars that are over \( B_j^{(k)} \).

**Example 2.2** The picture below represents a Bar Code

\[
\begin{array}{ccccccccc}
1 & & & & & & & & \\
2 & & & & & & & & \\
3 & & & & & & & &
\end{array}
\]

\(^1\) In this paper, the ordering on the variables will be fundamental. Unless otherwise specified we will consider \( x_1 < \ldots < x_n \).
Now we briefly sketch how to construct a Bar Code associated to a finite set of terms, following\(^2\) [6].

We start considering a term \(t = x_1^{\beta_1} \cdots x_n^{\beta_n} \in \mathcal{T} \subset k[x_1, \ldots, x_n]\) and defining the term \(\pi^i(t) := x_1^{\beta_1} \cdots x_n^{\beta_n} \in \mathcal{T}\), for each \(i \in \{1, \ldots, n\}\). We can define \(\pi^i(t), i \in \{1, \ldots, n\}\), for each term \(t\) of an ordered set of terms \(M = \{t_1, \ldots, t_m\}\), in particular ordered increasingly with respect to \(\text{Lex}\), getting \(\overline{M}^{[i]} := \{\pi^i(t)|t \in M\}\).

We point out that, while in \(M\) there are no repeated elements, they can occur in any of the lists \(\overline{M}^{[i]}\). If it happens, the repeated elements are clearly adjacent, being each \(\overline{M}^{[i]}\) a lexicographically ordered list. We continue computing the \(n \times m\) matrix of terms \(M\), defined so that its \(i\)-th row is \(\overline{M}^{[i]}\), \(i = 1, \ldots, n\).

**Definition 3.3** The Bar Code diagram \(B\) associated to \(M\) (or, equivalently, to \(\overline{M}\)) is a \(n \times m\) diagram, made by segments such that the \(i\)-th row of \(B\), \(1 \leq i \leq n\) is constructed as follows:

1. take the \(i\)-th row of \(M\), i.e. \(\overline{M}^{[i]}\)
2. consider all the sublists of repeated terms, i.e. \([\pi^i(t_{j_1}), \pi^i(t_{j_1+1}), \ldots, \pi^i(t_{j_1+h})]\) such that \(\pi^i(t_{j_1}) = \pi^i(t_{j_1+1}) = \ldots = \pi^i(t_{j_1+h})\), noting that\(^3\) \(0 \leq h < m\)
3. underline each sublist with a segment
4. delete the terms of \(\overline{M}^{[i]}, 2 \leq i \leq n\), leaving only the segments (i.e. the \(i\)-bars).

Each 1-bar \(B_j^{(1)}, j \in \{1, \ldots, \mu(1)\}\) remains labeled with the term \(t_j \in \overline{M}^{[1]}\).

We point out that a Bar Code diagram satisfies the condition of Definition 2.1 so it is a Bar Code.

**Example 2.4** From the set \(M = \{x_1, x_1^2, x_1^3 x_2, x_3, x_1 x_3, x_1^2 x_3\}\), we get the Bar Code of Example 2.2.

Bar Codes have been implemented in C and the implementation discussed in detail, also with a testing, in [11].

Essentially we use lists that are linked by pointers. Three different lists are needed: one for containing the monomials, one for the single bars and finally one for the levels of the Bar Code, which represent the variables.

### 3 Janet Decomposition and Completeness

Janet, in [34], considered a monomial/semigroup ideal \(J \subset \mathcal{T}\) and the related monomial basis \(G(J)\), introduced the concept of *multiplicative variable*, as well as the decomposition of \(J\) into disjoint *cones*, characterizing what Gerdt-Blinkov would have called an *involutive division*.

**Definition 3.1** [34, ppg.75-9] Let \(U \subset \mathcal{T}\) be a set of terms and \(t = x_1^{\alpha_1} \cdots x_n^{\alpha_n}\) be an element of \(U\). A variable \(x_j\) is called multiplicative for \(t\) with respect to \(U\) if there is no term in \(U\) of the form \(t' = x_1^{\beta_1} \cdots x_j^{\beta_j} x_{j+1}^{\alpha_{j+1}} \cdots x_n^{\alpha_n}\) with \(\beta_j > \alpha_j\). We denote by \(M(t, U)\) the set of multiplicative variables for \(t\) with respect to \(U\).

The variables that are not multiplicative for \(t\) with respect to \(U\) are called *non-multiplicative* and we denote by \(N M(t, U)\) the set containing them.

Once stated this definition, we can explicit the divisibility relation characterizing the involutive division. Given \(t \in U\) and \(w \in \mathcal{T}, t | w\) with respect to Janet division, if and only if

\[ w = t v \text{ and } \forall x_j \mid v, \ j \in \{1, \ldots, n\}, x_j \in M(t, U). \]

The term \(t\) is called *involutive divisor* of \(w\) with respect to Janet division and we will write \(t \mid_J w\).

**Definition 3.3** is dependent on the variables ordering, as shown in the following example.

**Example 3.2** Let us consider a simple example in two variables, letting \(U = \{x_1^2 x_2, x_2^2\} \subset k[x_1, x_2]\). If \(x_1 < x_2\), then we have that \(M(x_1^2 x_2, U) = \{x_1\}\) and \(N M(x_1^2 x_2, U) = \{x_2\}\); for the second term all variables are multiplicative, so \(M(x_2^2, U) = \{x_1, x_2\}, N M(x_2^2, U) = \emptyset\). If, instead \(x_2 < x_1\), then \(M(x_1^2 x_2, U) = \{x_1, x_2\}, N M(x_1^2 x_2, U) = \emptyset\) and \(M(x_2^2, U) = \{x_2\}, N M(x_2^2, U) = \{x_1\}\). This shows the dependency on the order of the variables.

\(^2\) An alternative construction has been given in detail in [5].

\(^3\) If a term \(\pi^i(t_j)\) presents no repetition in \(\overline{M}^{[i]}\), the sublist containing it will contain only that term, so \(h = 0\).
Definition 3.3 With the previous notation, the cone of $t$ with respect to $U$ is the set

$$C(t, U) := \{tx_1^{\lambda_1} \cdots x_n^{\lambda_n} \mid \lambda_j \neq 0 \text{ only if } x_j \in M(t, U)\}.$$ 

Example 3.4 ([7]). For the set $U = \{x_1x_3, x_2x_3\} \subseteq k[x_1, x_2, x_3]$, we have $M(x_1x_3, U) = \{x_1, x_3\}$ and $M(x_2x_3, U) = \{x_1, x_2, x_3\}$, so $C(x_1x_3, U) = \{x_1^i x_3^j : i, j \in \mathbb{N} \setminus \{0\}\}$ and $C(x_2x_3, U) = \{tx_2x_3 : t \in T\}$.

Note that we have $C(t, U) \cap U = \{t\}$, by definition of multiplicative variable.

In [34], Janet introduced also the notion of complete system, together with the completion procedure, namely the procedure to produce a complete system with respect to the decomposition in cones. Clearly, also the completion is dependent on the variables ordering.

Definition 3.5 [34, ppg.75-9] A set of terms $U \subset T$ is called complete if for every $t \in U$ and $x_j \in NM(t, U)$, there exists $t' \in U$ such that $x_jt \in C(t', U)$, namely there is an involutive divisor of $x_jt$ with respect to Janet division.

Remark 3.6 If $U \subseteq k[x_1 \ldots x_n]$ has cardinality one, then it is always a complete set. Indeed, its unique element has no non-multiplicative variables.

Also the order ideal $NW(J)$, associated to a monomial/semigroup ideal $J$, can be decomposed into disjoint cones. This decomposition has been introduced in [34], where Janet was describing Riquier’s idea to represent the general solution of a PDE problem [42].

Given a finite set of terms $U \subset T \subset k[x_1, \ldots, x_n]$, $x_1 < x_2 < \ldots < x_n$, it is possible to associate to it a Bar Code $B$. The diagram allows to read directly which are the Janet-multiplicative variables of the terms in $U$. First (see [6,7]), we place a star symbol $*$ in the following positions:

a) on the right of $B^{(i)}_{\mu(i)}$, $\forall 1 \leq i \leq n$;

b) between two consecutive bars $B^{(i)}_j$ and $B^{(i)}_{j+1}$ not lying over the same $(i+1)$-bar, $\forall 1 \leq i \leq n-1$, $\forall 1 \leq j \leq \mu(i) - 1$.

Consider then a term $t \in U$; to detect its multiplicative variables we only have to check the presence/absence of stars just after bars over which $t$ lies, as stated in the following proposition (see [7] for its proof).

Proposition 3.7 Let $U \subseteq T$ be a finite set of terms and let us denote by $B$ its Bar Code. For each $t \in U$, $x_i$, $1 \leq i \leq n$, is multiplicative for $t$ if and only if, in $B$, the $i$-bar $B^{(i)}_j$, over which $t$ lies, is followed by a star.

Example 3.8 Let us consider the set $U = \{x_1, x_1^2, x_2, x_1x_3\} \subseteq k[x_1, x_2, x_3]$, with $x_1 < x_2 < x_3$. The corresponding Bar Code is

\[
\begin{array}{cccc}
0 & x_1 & x_1^2 & x_2 & x_1x_3 \\
1 & \text{---} & \text{---} & \ast & \ast & \ast \\
2 & \text{---} & \text{---} & \ast & \ast & \ast \\
3 & \text{---} & \text{---} & \text{---} & \ast & \ast
\end{array}
\]

According to Proposition 3.7, $x_1$ has no multiplicative variables. Indeed, following Definition 3.1, $x_1$ is not multiplicative since $x_1^2 \in U$, $x_2$ is not multiplicative since $x_2 \in U$ and $x_3$ is not multiplicative since $x_1x_3 \in U$. Therefore, $C(x_1, U) = \{x_1\}$.

Remark 3.9 Bar Code - which, in principle can be used for different purposes in various contexts, can be used to see what are the multiplicative variables of the terms in a finite set. This makes the Bar Code, in this specific context, another way to formulate Gerdt-Blinkov-Yanovich Janet tree [26]. Indeed [7] devotes its last section to a

---

4 In [6,7] a set of terms is constructed with an analogous procedure; the paper [10] links this set to Pommaret bases [40,41].
comparison between them. The way we develop our construction makes the Bar Code similar to another equivalent formulation of the tree, the one by [45]. Of course all presentations allow to deduce a Janet decomposition and the two above representations lead to algorithms that differ from those coming from Proposition 3.7. This is also pointed out in [7], that the interested reader can see for more details.

Completeness of a given finite set \( U \) can be detected by means of the Bar Code, as stated in the following proposition, which specializes [8, Thm. 11], which is instead proved in the context of Janet-like divisions [24,25].

**Proposition 3.10** Let \( U \subseteq \mathcal{T} \) be a finite set of terms and \( \mathcal{B} \) be its Bar Code. Let \( t \in U, x_i \in NM(t, U) \) and \( \mathcal{B}^{(i)}_j \) the i-bar under \( t \). Let \( s \in U \); it holds \( s \mid_j x_i t \) if and only if

1. \( s \mid x_i t \)
2. \( s \) lies over \( \mathcal{B}^{(i)}_{j+1} \)
3. for each variable \( x_j \) appearing with nonzero exponent in \( \frac{x_i t}{s} \) there is a star after the \( j \)'-bar under \( s \).

From Proposition 3.10 we finally get the following theorem.

**Theorem 3.11** Let \( U \subseteq \mathcal{T} \) be a finite set of terms and \( \mathcal{B} \) be its Bar Code. Then \( U \) is a complete set if and only if for each \( t \in U \) and each \( x_i \in NM(t, U) \), called \( \mathcal{B}^{(i)}_j \) the i-bar under \( t \), there exists a term \( s \in U \) satisfying Conditions 1, 2, 3 of Proposition 3.10.

Proposition 3.10 and Theorem 3.11 tell us that taken a finite \( U \subseteq \mathcal{T} \), we can use the Bar Code to check whether \( U \) is complete or not. For each \( t \in U \) and each \( x_i \in NM(t, U) \), consider the i-bar \( \mathcal{B}^{(i)}_j \), \( 1 \leq j \leq \mu(i) \) under \( t \); we search an involutive divisor among the terms over \( \mathcal{B}^{(i)}_{j+1} \), checking Conditions 1,3 above. We see now two simple examples of this procedure.

**Example 3.12** Consider the set \( U = \{ x_1^3, x_2^3, x_2^4 x_3, x_3^2 \} \subseteq k[x_1, x_2, x_3] \) and its Bar Code

\[
\begin{array}{cccccc}
0 & x_1^3 & x_2^3 & x_2^4 x_3 & x_3^2 \\
1 & \star & \star & \star & \star & \star \\
2 & \star & \star & \star & \star & \star \\
3 & \star & \star & \star & \star & \star \\
\end{array}
\]

Take \( t = x_1^3 \) and \( x_2 \in NM(t, U) = \{ x_2, x_3 \} \); \( t \) lies over \( \mathcal{B}^{(2)}_1 \) and the only term over \( \mathcal{B}^{(2)}_2 \) is \( x_2 \mid x_1^3 x_2 = tx_2 \), so \( tx_2 \) has no involutive divisor on \( U \) and this implies that our set is actually non-complete.

**Example 3.13** Consider the set \( U = \{ x_1^2, x_1 x_2 \} \subseteq k[x_1, x_2], x_1 < x_2 \). Its Bar Code is as follows.

\[
\begin{array}{cccc}
0 & x_1^2 & x_1 x_2 \\
1 & \star & \star & \star \\
2 & \star & \star & \star \\
\end{array}
\]

Looking at the stars, we can deduce \( M(x_1^2, U) = \{ x_1 \}, NM(x_1^2, U) = \{ x_2 \} \), \( M(x_1 x_2, U) = \{ x_1, x_2 \}, NM(x_1 x_2, U) = \emptyset \). Now, \( t = x_1^2 \) lies over \( \mathcal{B}^{(1)}_1 \) and over \( \mathcal{B}^{(1)}_2 \) there is only \( x_1 x_2 \) such that \( x_1 x_2 \mid x_1^2 x_2 \).

Since \( x_1 \in M(x_1 x_2, U), x_1 x_2 \mid x_1^2 x_2 \) and we can conclude that \( U \) is complete, with respect to the given ordering on the variables.

4 A “Greedy” Algorithm for Complete Sets

In this section, given a finite set of terms \( U = \{ t_1, \ldots, t_m \} \subseteq \mathcal{T} \), we try to find out if there is an ordering on the variables \( x_1, \ldots, x_n \) that make \( U \) complete with respect to that. As explained in Sect. 3, the Bar Code allows to detect the completeness of \( U \). The involved construction depends on the ordering of the variables, since the Janet multiplicative variables depend on such an ordering. Thus, if we want to solve the problem, in principle, we would need to construct \( n! \) different Bar Codes and check all of them. Of course, it is desirable to find a more efficient way to do that. We show now that we can look for the solution of our problem in a “greedy” way, so that we can avoid most of the tests.\(^5\) In order to do so, we first come back to [34].

\(^5\) The use of the word “greedy” is actually not the Computer Theoretic one, we only intend that we try to make choices that try to decrease the number of tests.
Let $U = \{t_1, \ldots, t_m\} \subseteq \mathcal{T}$ be a finite set of terms, $t_i = x_1^{a_{i1}} \cdots x_n^{a_{in}}$ and $t_i' = x_1^{a_{i1}} \cdots x_{n-1}^{a_{i,n-1}} = t_i / x_n^{a_{in}}$, for $i = 1, \ldots, m$. Let $\alpha = \max\{a_{n,i}, 1 \leq i \leq m\}$. For each $\lambda \leq \alpha$, we define $I_\lambda := \{i : 1 \leq i \leq m | a_{n,i} = \lambda\}$, the set indexing the terms in $U$ with $n$-th degree equal to $\lambda$, and $U'_\lambda := \{t_i' | i \in I_\lambda\} \subset \mathcal{T}[\{1, \ldots, n-1\}]$. Being $U'_\lambda$ a finite set of terms in $\mathcal{T}[\{1, \ldots, n-1\}]$, we can define Janet division on it and we can observe that, by Definition 3.1, 

- for each $t_i' \in U'_\lambda$, $M(t', U) \cap \{x_1, \ldots, x_{n-1}\} = M(t', U'_\lambda)$;
- if $t = x_1^{a_1} \cdots x_n^{a_n} \in U$, $U'$ is complete and $a_n < \alpha$, then $x_n \in NM(t, U)$ and the involutive divisor of $x_n t$ according to Janet division is a term $s \in U$ such that $s' \in U'_{\lambda+1}$.

These are the main observations leading to the following Proposition, first stated in [34] and then explicitly proved in [37].

**Proposition 4.1** ([34,37]). With the notation above, $U$ is complete if and only if the two conditions below hold:

1. For each $\lambda \in \{a_{n,i}, 1 \leq i \leq m\}$, $U'_\lambda$ is a complete set;
2. $\forall t_i' \in U'_\lambda$, $\lambda < \alpha$, there exists $j \in \{1, \ldots, m\}$ such that
   a) $t_j' \in U''_{\lambda+1}$;
   b) $t_j' \in C(t_j', U''_{\lambda+1})$.

Now, using the Bar Codes and the above Proposition 4.1, we can check whether there is an ordering on the variables making a given set $U$ complete.

Given the set $U = \{t_1, \ldots, t_m\} \subseteq \mathcal{T}$, what we want to do in the algorithm is constructing the Bar Code $B$ of $U$ with the ordering that makes $U$ complete (and return failure if it is impossible). In particular, what we do is starting choosing a candidate for the maximal variable, so we first construct the bars on the bottom of $B$, and we proceed until we get to the minimal one. The choices we make on the variables impose conditions that come from Proposition 3.10, therefore, for each variable choice we make at each step, we need to check whether the conditions we have imposed in the previous steps are satisfied or not. In case some of them are not, we have to revoke the last made choice and go back. Therefore, one bar after the other, conditions on which variables should be multiplicative for a certain term in order for it to be an involutive divisor are imposed and checked in the following steps.

In particular, this is made easy by the use of the Bar Code as a sort of terms’ storage. To be more precise, with the Bar Code one can store the exponents of each variable, for all terms in $U$, and their reciprocal relations in the sense that:

**two terms are on the same bar, on the rows $n, n-1, \ldots, i$, and on different $(i-1)$-bars if and only if they have the same exponents in $x_n, \ldots, x_i$, but they differ in the exponent of $x_{i-1}$.**

The subdivision of terms on the basis of their exponents is exactly the one required by Proposition 3.10, so a Bar Code is a way to store the terms in $U$ that is especially suitable for the tests required by Proposition 3.10.

We describe now the algorithm, and then we will display some examples. The main procedure is ORDERING (Algorithm 1) while Algorithms 2,3,4,5 are the subroutines on which it depends and the reader can find their pseudocode displayed in Appendix A.

Let $X = \{x_1, \ldots, x_n\}$ be the set of all variables. First of all, the procedure ORDERING computes the subset $Y \subseteq X$ of good candidates for being the maximal variable, scanning the elements of $X$ (Algorithm 1, line 2). In particular, it relies on the subroutine CANDIDATES (Algorithm 3) for this task.

The procedure CANDIDATES considers the set of all variables and deletes all those that are not good candidates for being the maximal one. It takes as input a list of terms $L$ (in this case $L = \{U\}$, so $|L| = 1$ and $L[1] = 1$) and a list of variables $C$ (in this case $C = \{x_1, \ldots, x_n\}$) and returns the list of good candidates for being the maximal one.
Algorithm 1 Ordering

1: procedure ORDERING($M, X$) \(\triangleright\) $M$ is a given list of terms; $X$ is the list of all variables
2: \hspace{1em} $Y :=$CANDIDATES($M, X$)
3: \hspace{1em} for $x_i \in Y$ do
4: \hspace{2em} $A = \{A_i^{(i)}\}$, $1 \leq j \leq \mu(i)$
5: \hspace{2em} \hspace{1em} $T :=$FRIENDS($A, X \setminus \{x_i\}, x_i, \emptyset$)
6: \hspace{2em} \hspace{1em} if $T \neq \emptyset$ then
7: \hspace{3em} \hspace{1em} \hspace{1em} continue
8: \hspace{2em} \hspace{1em} end if
9: \hspace{2em} \hspace{1em} if $\left| A_i^{(i)} \right| = 1, 1 \leq j \leq \mu(i)$ then \(\triangleright\) Unitary case.
10: \hspace{2em} \hspace{1em} \hspace{1em} $ord =$Append($X \setminus \{x_i\}, x_i$) \(\triangleright\) We append $x_i$ at the end of the list $X \setminus \{x_i\}$, namely the list $X$ from which $x_i$ has previously been pruned. This way $x_i$ is the last (and so maximal) variable.
11: \hspace{2em} \hspace{1em} \hspace{1em} return $ord$
12: \hspace{2em} \hspace{1em} end if
13: \hspace{2em} \hspace{1em} $C :=$COMMON($A, X, x_i, T$)
14: \hspace{2em} \hspace{1em} if $C \neq \emptyset$ then
15: \hspace{2em} \hspace{2em} $ord =$Append($C, x_i$)
16: \hspace{2em} \hspace{1em} \hspace{1em} return $ord$
17: \hspace{2em} \hspace{1em} \hspace{1em} else continue
18: \hspace{2em} \hspace{1em} end if
19: \hspace{2em} \hspace{1em} end for
20: \hspace{2em} return $\emptyset$
21: end procedure

In order to decide whether a variable is a good candidate or not, CANDIDATES scans $L$ and, for each list of terms in $L$, applies the subroutine CANDIDATEVAR (Algorithm 3, lines 2-4), returning the good candidates in Algorithm 3, line 5. For $i = 1, \ldots, |C|$, CANDIDATEVAR computes the sets $D_i := \{\beta \in \mathbb{N} | \exists t \in U. \ deg_t(t) = \beta\}$ (Algorithm 2, lines 3-4) and excludes from the good candidates all variables $x_{\bar{i}}$, $\bar{i} \in \{1, \ldots, |C|\}$, for which there exists $\gamma \in D_{\bar{i}}$ such that $\gamma < \max(D_{\bar{i}})$ and $\gamma + 1 \notin D_{\bar{i}}$ (Algorithm 2, lines 5-6). This procedure is justified by the following lemma, strongly depending on Proposition 4.1.

Lemma 4.2 Let $U \subseteq \mathcal{T}$ be a finite set of terms. For $i = 1, \ldots, n$, consider the sets $D_i := \{\deg_t(t) : t \in U\}$. Suppose that for some $\bar{i}, \bar{i} \in \{1, \ldots, n\}$, there exists $\gamma \in D_{\bar{i}}$ such that $\gamma < \max(D_{\bar{i}})$ and $\gamma + 1 \notin D_{\bar{i}}$. Then $U$ is not a complete set for any ordering on the variables with $x_{\bar{i}}$ as maximal variable.

Proof In order to be complete, the set $U$ should satisfy the conditions of Proposition 4.1. Under our hypotheses, we have $\alpha > \gamma + 1$, $I_{\gamma} \neq \emptyset$ and $I_{\gamma+1} = \emptyset$. To each $i \in I_{\gamma}$ corresponds a term $t'_{i} \in U'_{\gamma}$ and $j \in \{1, \ldots, |U|\}$ such that $t'_{i} \in C(t'_{j}, U)$ and $t'_{j} \in U'_{\gamma+1}$ must exist, but this is impossible since, being $I_{\gamma+1} = \emptyset$, also $U'_{\gamma+1} = \emptyset$. \(\square\)

After the routine CANDIDATE concludes its work at line 2 of Algorithm ORDERING, the subset $Y \subseteq X$ of good candidates for being the maximal variable is produced. It may happen that $Y$ is non empty, but also that no variables have been selected to be in there.

In the (unfortunate but easy) case $Y = \emptyset$, no variable is suitable for being the maximal one and making $U$ complete; this implies that $U$ is not complete for any ordering on the variables (Algorithm 1, line 20).

For an example of this case, see Example 4.5.

Suppose now $\emptyset \neq Y \subseteq X$; ORDERING picks a variable $x_i \in Y$ and considers it as the maximal variable (Algorithm 1, line 3). Then it starts the construction of the last row\(^6\) of the Bar Code associated to $U$ (Algorithm 1, line 4). In particular, the elements of $U$ are rearranged increasingly with respect to their $i$-degree, imposing, in addition, $t < t'$ when $t \mid t'$ for some $t, t' \in U$ with $\deg_t(t) = \deg_t(t')$. Denote the $i$-degrees of the terms in $U$

---

\(6\) Remember that the last row of a Bar Code, namely that on the bottom, is the row associated to the maximal variable (see Sect. 2.2).
by \(\lambda_1 < \lambda_2 < \ldots < \lambda_{\mu(i)}, 1 \leq \mu(i) \leq |U|\). The \(i\)-bars are \(B_1^{(i)}, \ldots, B_r^{(i)}\) and they are drawn under the terms, grouping them according to their \(i\)-degree: the terms of \(i\)-degree \(\lambda_1\) are underlined by \(B_1^{(i)}\), those of \(i\)-degree \(\lambda_2\) by \(B_2^{(i)}\) and so on (see Example 4.6).

By construction, then, there is an obvious bijection \(\phi_j^{(i)}\) between the set \(A_j^{(i)}\) of terms over \(B_j^{(i)}, 1 \leq j \leq \mu(i)\) and the set \(U_{\lambda_j}^{(i)}\) of Proposition 4.1.

Afterwards, ORDERING (Algorithm 1 line 5) launches the subroutine FRIENDS (Algorithm 4), whose aim is essentially to mimic Condition 2 of Proposition 4.1. In particular, it adds requirements on what variables should be multiplicative for the terms to make Condition 2 hold and it checks whether the requirements on the variables imposed by its previous executions are met.

This is the first time in which FRIENDS is run by the algorithm, so there are no previously imposed requirements on the multiplicative variables (we only have imposed \(x_i\) as maximal variable) so the part of FRIENDS that checks such requirements (Algorithm 4, lined 11 - 31) is skipped for now. It only lists the requirements imposed by the choice of \(x_i\) as maximal variable, so that Condition 2. of Proposition 4.1 holds.

For each \(1 \leq j < \mu(i)\), consider the bar \(B_j^{(i)}\) and the corresponding set of terms \(A_j^{(i)}\).

For each \(t \in A_j^{(i)}\), FRIENDS computes the set \(U(t, x_j) = \{(u, V) \mid u \in A_j^{(i)}\} \text{ and } V : tx_i = um, m \in T[V]\). The elements in \(U(t, x_j)\) are the candidates for being the involutive divisor of \(tx_i\) (or, in other words, are the candidate terms for satisfying Condition 2 of Proposition 4.1) while \(V\) is the set of variables – smaller than \(x_i\) – that must belong to \(M(u, U)\) for \(u\) being the involutive divisor of \(tx_i\) (Algorithm 4, lines 2–10). Indeed, due to the bijections \(\phi_j^{(i)}: \phi_j^{(i+1)}: t' := t/x_j^{\lambda_j} U_{\lambda_j}^{(i)}\) and \(u' := u/x_j^{\lambda_j+1} U_{\lambda_j+1}^{(i)}\) and, if the variables in \(V\) are multiplicative for \(u\), then \(t' \in C(u', U_{\lambda_j+1}^{(i)})\). If one of the sets \(U(t, \{x_i\})\) is empty, then there is no candidate for being the involutive divisor of \(tx_i\) if \(t\) is not a good choice for being the maximal variable. In this case, we consider \(Y\) again and we restart with a new maximal variable (Algorithm 1, lines 6 - 8). See Example 4.7 in which such process is shown.

Suppose to be in the non-failure case. If, in addition, for \(1 \leq j < \mu(i)\) over the bar \(B_j^{(i)}\) there is only one term, we say that all the bars are unitary: we are in the unitary case (Algorithm 1, lines 9 - 12). In this special case, all variable orderings having \(x_i\) as maximal variable make \(U\) a complete set of terms.

Indeed, in this case, for each choice on the ordering of the following (and so, smaller) variables, their corresponding bars will be unitary again and, by the construction of the stars, all of them will be followed by a star. In other words, for each \(t \in U\), and for each \(x_j \neq x_i, x_j \in M(t, U)\). Moreover, for each \(t \in U, |U(t, \{x_i\})| = 1\), so let \((u, V)\) be the only element in \(U(t, x_i)\), then \(x_i \notin V\), so all variables in \(V\) are multiplicative for \(u\) and this makes \(u\) the required involutive divisor of \(tx_i\), ensuring the completeness of \(U\) (see Example 4.8).

If the unitary case does not occur (Algorithm 1 line 13), we have to consider the next variable and continue drawing a new level of the Bar Code, using the routine COMMON (Algorithm 5).

To get the candidates for being the next variable, we execute the procedure CANDIDATEVAR to each \(i\)-bar and (procedure CANDIDATES) we intersect the results. Indeed, we want that the next variable is a good candidate to make all the sets \(U_{\lambda_i}\) complete, according to Condition 1. of Proposition 4.1.

If the intersection is empty (Algorithm 5, lines 4 - 5) there are no such good candidates, then \(x_i\) was not a good choice for being the maximal variable and we have to come back and repeat the whole procedure for another maximal variable (Algorithm 1 line 17).

Otherwise, we pick some \(x_i\) in the intersection (Algorithm 5 line 6), and for each \(1 \leq j \leq \mu(i)\), we order the terms over \(B_j^{(i)}\) in increasing order according to the \(l\)-degree, imposing in addition \(t < t'\) when \(t \not| t'\) for some \(t, t' \in U\) with \(deg_l(t) = deg_l(t')\) (exactly as we did for \(x_i\)). Then we draw all the \(l\)-bars (Algorithm 5 line 8).

Employing again the routine FRIENDS, separately for each \(i\)-bar, we look for candidate involutive divisors for the cases in which \(x_i\) is not multiplicative (Algorithm 4, lines 2 - 10). Moreover, we check whether the choice of \(x_i\) is a good one, by checking that \(x_i\) is multiplicative for all terms we imposed it to be in the previous application of the routine FRIENDS (Algorithm 4, lines 11 - 31).

More precisely, for each \(t\) over \(B_j^{(i)}\), \(1 \leq j < \mu(i)\), we have constructed a set \(U(t, \{x_i\})\) of candidate involutive divisors for \(tx_i\). Given \((u, V) \in U(t, x_i)\), if \(x_i \notin V\), then the multiplicity of \(x_i\) is irrelevant for \(u\), so \((u, V)\) still
remains a good candidate for being an involutive divisor. It is still a good candidate also if \( x_t \in V \) and the \( l \)-bar of \( u \) is in one of the conditions for being followed by a star (see sect. 2.2), since it means that \( x_t \) is multiplicative for \( u \). Otherwise, we remove \((u, V)\) from the candidates. If for some \( t \), its candidate list has no elements, \( x_t \) was not a good choice and so we come back to pick another variable in place (Algorithm 5 line 11).

If FRIENDS gives a positive outcome, then a new variable has been placed and the routine COMMON keeps calling itself until (Algorithm 1, lines 13 - 21)

- all variables have been settled (positive outcome)
- we get to the unitary case (positive outcome)
- continue revocations of choices lead to failure, in the sense that there are no more variables to pick (negative outcome, there is no ordering on the variables making the set complete).

Termination and correctness of Algorithm 1 come from Proposition 4.1, as it is proved in the following corollary.

**Corollary 4.3** Algorithm 1 terminates in a finite number of steps and it is correct.

**Proof** The algorithm terminates since there is a finite number of variables and each time we pick a variable as candidate, we remove it from the candidates’ list, so we do not choose a variable in some position of the ordering more than once.

The correctness, instead, is an easy consequence of Proposition 4.1, since the algorithm executes the tests imposed by that proposition. \( \square \)

**Remark 4.4** We point out that, even in the case in which the given set \( U \) is not complete for any variables ordering, it is possible to store the state in which most of the variables have been settled, before retracting due to some failure condition. This - though not being a warranty of minimality for the terms one has to add in order to get the completion - can reduce the number of tests one has to do in the first step of completion.

We conclude this section with some examples that focus on specific parts of the algorithm, in order to explain them better. In particular, the following example displays a very small set that can never be complete, for any ordering one imposes on the variables.

**Example 4.5** Consider \( U = \{x_1 x_2^2, x_1 x_2\} \subset k[x_1, x_2] \). Such a set is not complete since \( D_1 = D_2 = \{1, 3\} \). As a confirmation, we can see that, if \( x_1 < x_2 \), we have

\[
\begin{array}{c|c|c}
0 & x_1^3 x_2 & x_1 x_2^3 \\
1 & * & * \\
2 & * & *
\end{array}
\]

Then \( M(x_1^3 x_2, U) = \{x_1\} \), \( M(x_1 x_2^3, U) = \{x_1, x_2\} \) and \( x_1^3 x_2^3 \) does not belong neither to \( C(x_1^3 x_2, U) \) nor to \( C(x_1 x_2^3, U) \).

On the other hand, if \( x_2 < x_1 \), we have

\[
\begin{array}{c|c|c}
0 & x_1 x_2^3 & x_1^3 x_2 \\
2 & * & * \\
1 & * & *
\end{array}
\]

Thus \( M(x_1 x_2^3, U) = \{x_2\} \), \( M(x_1^3 x_2, U) = \{x_1, x_2\} \) and \( x_1^3 x_2^3 \) does not belong neither to \( C(x_1^3 x_2, U) \) nor to \( C(x_1 x_2^3, U) \).

In the next example, we show the choice of maximal variables.

**Example 4.6** Let us consider the set \( U = \{x_1, x_1^2, x_2, x_1 x_3\} \subset k[x_1, x_2, x_3] \); we first compute \( D_1 = \{0, 1, 2\} \), \( D_2 = D_2 = \{0, 1\} \). All the variables are good candidates for being the maximal one. We pick, for example, \( x_3 \), so we have

\[
\begin{array}{c|c|c|c|c}
0 & x_1 & x_1^2 & x_2 & x_1 x_3 \\
3 & * & * & * & *
\end{array}
\]
λ₁ = 0, λ₂ = 1, so \(A_1^{(3)} = U_0 = \{x_1, x_1^2, x_2\}\) and \(A_2^{(3)} = U_1 = \{x_1x_3\}\).

We remark that we could also have picked another variable, obtaining a different Bar Code; for example, picking \(x_1\), we would have got:

\[
\begin{array}{cccc}
0 & x_2 & x_1 & x_1x_3 & x_1^2 \\
1 & & & & *
\end{array}
\]

We continue with the same example and we show the tests on the variable’s choice, retracting back once.

**Example 4.7** Coming back to Example 4.6, we have \(U(x_1, x_3) = \{(x_1x_3, \emptyset)\}\), \(U(x_1^2, x_3) = \{(x_1x_3, \{x_1\})\}\) and \(U(x_2, x_3) = \emptyset\), so \(x_3\) was a bad choice for being the maximal variable. We try with \(x_1\), getting

\[
\begin{array}{cccc}
0 & x_2 & x_1 & x_1x_3 & x_1^2 \\
1 & & & & *
\end{array}
\]

Now, \(U(x_2, x_1) = \{(x_1, \{x_2\})\}\), \(U(x_1, x_1) = \{(x_1^2, \emptyset)\}\) and \(U(x_1x_3, x_1) = \{(x_1^2, \{x_3\})\}\), so \(x_1\), at least for now, is a good choice for the maximal variable.

We show now the unitary case.

**Example 4.8** For \(U = \{x_1^3, x_1x_2, x_2^2\} \subset k[x_1, x_2]\), \(D_1 = \{0, 1, 3\}\) and \(D_2 = \{0, 1, 2\}\), so \(Y = \{x_2\}\). Chosing \(x_2\) as maximal variable the Bar Code becomes:

\[
\begin{array}{cccc}
0 & x_1^3 & x_1x_2 & x_2^2 \\
1 & & & & \\
2 & & & & 
\end{array}
\]

and we have \(U(x_1^3, x_2) = \{(x_1x_2, \{x_1\})\}\), \(U(x_1x_2, x_2) = \{(x_2^2, \{x_1\})\}\). All the 2-bars are unitary, so, completing the Bar Code we get

\[
\begin{array}{cccc}
0 & x_1^3 & x_1x_2 & x_2^2 \\
1 & * & * & * & * \\
2 & * & * & * & *
\end{array}
\]

We can easily check from the diagram that \((x_1^3) \cdot x_2 \in C(x_1x_2, U)\) and \((x_1x_2) \cdot x_2 \in C(x_2^2, U)\). Therefore \(U\) is complete.

We conclude now Examples 4.6 and 4.7.

**Example 4.9** From

\[
\begin{array}{cccc}
0 & x_2 & x_1 & x_1x_3 & x_1^2 \\
1 & & & & *
\end{array}
\]

we choose now \(x_2\) as following variable and we get
Example 4.10

Consider the set
\[ 0 \ x_2 \ x_1 \ x_1x_3 \ x_1^2 \]

\[ 2 \quad * \quad \quad * \quad \quad * \quad \quad * \]

\[ 1 \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad * \]

Now, running FRIENDS we do not impose any condition on the terms over the 2-bars. Indeed, over each 1-bar there is only one 2-bar and so, that 2-bar is followed by a star, this implying that \( x_2 \) is multiplicative for all terms. Moreover, we have only to check \( U(x_2, x_1) = \{(x_1, \{x_2\})\} \); being \( x_2 \in M(x_1, U) \), the procedure gives a positive outcome. Finally choosing \( x_3 \), we get

\[ 0 \ x_2 \ x_1 \ x_1x_3 \ x_1^2 \]

\[ 3 \quad * \quad \quad * \quad \quad * \quad \quad * \]

\[ 2 \quad * \quad \quad * \quad \quad * \quad \quad * \]

\[ 1 \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad * \]

Now, \( x_3 \) is multiplicative for \( x_1^2 \) as required by \( U(x_1x_3, x_1) \) and we have \( U(x_1, x_3) = \{(x_1, \emptyset)\} \), so \( U \) turns out to be complete with the variables ordering \( x_3 < x_2 < x_1 \).

We point out that this is not the only ordering making \( U \) complete, in particular, for \( x_1 < x_3 < x_2 \) \( U \) is complete again:

\[ 0 \ x_1 \ x_2^2 \ x_1x_3 \ x_2 \]

\[ 1 \quad \quad \quad * \quad \quad * \quad \quad * \quad \quad * \]

\[ 3 \quad \quad \quad * \quad \quad * \quad \quad * \quad \quad * \]

\[ 2 \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad * \]

Indeed

- \( M(x_1, U) = \emptyset \), \( NM(x_1, U) = \{x_1, x_2, x_3\} \), with \( x_2^2 \in C(x_1^2, U) \), \( x_1x_2 \in C(x_2, U) \), \( x_1x_3 \in C(x_1x_3, U) \);
- \( M(x_1^2, U) = \{x_1\} \), \( NM(x_1^2, U) = \{x_2, x_3\} \), with \( x_2^2, x_2 \in C(x_2, U) \), \( x_1^2x_3 \in C(x_1x_3, U) \);
- \( M(x_1x_3, U) = \{x_1, x_3\} \), \( NM(x_1x_3, U) = \{x_2\} \), with \( x_1x_2x_3 \in C(x_2, U) \);
- \( M(x_2, U) = \{x_1, x_2, x_3\} \), \( NM(x_2, U) = \emptyset \).

Finally, we see a complete example for the execution of the whole procedure.

**Example 4.10** Consider the set

\[ M = \{x_2x_3, x_1^2, x_2^2, x_1, x_2, x_1x_2, x_1x_3x_4, x_1^2x_4, x_4x_3, x_2^2x_4, x_1^2x_3, x_2x_3, x_4x_3, x_3^2\} \subset k[x_1, x_2, x_3, x_4]. \]

First, we compute \( D_1 = D_2 = D_3 = \{0, 1, 2\} \), \( D_4 = \{0, 1\} \), deducing that each variable is a good candidate for being the maximal one, so \( Y = \{x_1, x_2, x_3, x_4\} \). We choose, for example, \( x_3 \), getting

\[ x_1^2 \quad x_1x_3 \quad x_2^2 \quad x_1^2x_4 \quad x_1x_2x_4 \quad x_2^2x_4 \quad x_1^2x_3 \quad x_2x_3 \quad x_4x_3 \quad x_3^2 \]

\[ 3 \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad * \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad * \]

Now, running FRIENDS for the first time, we get

- \( U(x_1^2, x_3) = \{(x_1^2x_3, \emptyset)\} \);
- \( U(x_1x_2, x_3) = \{(x_2x_3, \{x_1\})\} \);
\[ U(x_1^2, x_3) = \{(x_2 x_3, \{x_2\})\}; \]
\[ U(x_1^2 x_4, x_3) = \{(x_1^2 x_3, \{x_4\}), (x_3 x_4, \{x_1\})\}; \]
\[ U(x_1 x_2 x_4, x_3) = \{(x_2 x_3, \{x_1, x_4\}), (x_3 x_4, \{x_1, x_2\})\}; \]
\[ U(x_2^2 x_4, x_3) = \{(x_2 x_3, \{x_2, x_4\}), (x_3 x_4, \{x_2\})\}; \]
\[ U(x_2 x_3, x_3) = \{(x_2^3, \{x_2\})\}; \]
\[ U(x_3 x_4, x_3) = \{(x_3^2, \{x_4\})\}. \]

The procedure gives a positive outcome, so, since we are not in the unitary case, we apply COMMON. All the variables are good candidates for being the second in order of magnitude and, for example, we choose \( x_4 \), getting:

\[
\begin{array}{cccccccc}
 x_1^2 & x_1 x_2 & x_2^2 & x_1^2 x_4 & x_1 x_2 x_4 & x_2^2 x_4 & x_1^2 x_3 & x_2 x_4 \\
 4 & 3 \\
\end{array}
\]

We have:

- \( U(x_1^2, x_4) = \{(x_1^2 x_4, \emptyset), (x_4, \{x_1\})\}; \)
- \( U(x_2^2, x_4) = \{(x_2^2 x_4, \emptyset)\}; \)
- \( U(x_1 x_2, x_4) = \{(x_1 x_2 x_4, \emptyset)\}; \)
- \( U(x_2, x_4) = \{(x_4, \{x_2\})\}. \)

We check that the choice of \( x_4 \) is suitable for the conditions imposed in the previous step:

- for \( U(x_1^2 x_4, x_3) = \{(x_1^2 x_3, \{x_4\}), (x_3 x_4, \{x_1\})\} \) note that \( x_1^2 x_3 \) does not lie on the rightmost 4-bar, so \( x_4 \) is not multiplicative. Since we have more than one term associated to \( x_1^2 x_4 \), we only delete \( x_1^2 x_3 \) and keep \( x_3 x_4 \).

The same argument holds for \( x_1 x_2 x_4, x_2^2 x_4 \).

- For \( U(x_3 x_4, x_3) = \{(x_3^2, \{x_4\})\} \), since \( x_3^2 \) lies on the rightmost 4-bar, \( x_3^2 \) passes the test, remaining a good candidate for being an involutive divisor.

So we have:

- \( U(x_1^2, x_3) = \{(x_1^2 x_3, \emptyset)\}; \)
- \( U(x_1 x_2, x_3) = \{(x_2 x_3, \{x_1\})\}; \)
- \( U(x_2^2, x_3) = \{(x_2 x_3, \{x_2\})\}; \)
- \( U(x_1^2 x_4, x_3) = \{(x_3 x_4, \{x_1\})\}; \)
- \( U(x_1 x_2 x_4, x_3) = \{(x_3 x_4, \{x_1, x_2\})\}; \)
- \( U(x_2^2 x_4, x_3) = \{(x_3 x_4, \{x_2\})\}; \)
- \( U(x_1^2 x_3, x_3) = \{(x_3^2, \{x_1\})\}; \)
- \( U(x_2 x_3, x_3) = \{(x_2^3, \{x_2\})\}; \)
- \( U(x_3 x_4, x_3) = \{(x_3^2, \{x_4\})\}; \)
- \( U(x_1^2, x_4) = \{(x_1^2 x_4, \emptyset)\}; \)
- \( U(x_2 x_4, x_3) = \{(x_2 x_4, \{x_1\})\}; \)
- \( U(x_1 x_2 x_4, x_3) = \{(x_3 x_4, \{x_1, x_2\})\}; \)
- \( U(x_2^2 x_4, x_3) = \{(x_3 x_4, \{x_2\})\}; \)
- \( U(x_1^2 x_3, x_3) = \{(x_3^2, \{x_1\})\}; \)
- \( U(x_2 x_4, x_3) = \{(x_3 x_4, \{x_2\})\}. \)

We continue choosing \( x_2 \) as next variable and we get:

\[
\begin{array}{cccccccc}
 x_1^2 & x_1 x_2 & x_2^2 & x_1^2 x_4 & x_1 x_2 x_4 & x_2^2 x_4 & x_1^2 x_3 & x_2 x_4 \\
 2 & 3 \\
\end{array}
\]

This way, all the 2-bars are unitary. We check on the 2-bars to have nonincreasing exponents for \( x_1 \) and this is true. Moreover, we check that \( x_2 \) is multiplicative where it is marked, i.e. for \( x_2 x_3, x_3 x_4 \) but it clearly holds. The set \( M \) is complete for \( x_1 < x_2 < x_4 < x_3 \) and its final Bar Code with respect to the chosen ordering is
Appendix A. List of all Procedures

Before listing all procedure, we recall that a Bar Code is given by a concatenation of lists via pointers. We have one list for the levels (i.e. the variables), one for the bars within any level and finally one for the terms. Constructing a bar, then, means adding a new element to the list of bars, connecting it in the right position by means of pointers.

When we put a star at the end of a bar, we are putting a star symbol at the end of the corresponding bar, therefore we suppose known a procedure Star($x_i, t$), which takes a variable $x_i$ and a monomial $t$ as input and returns true if at level $i$, the bar under $t$ (therefore placed in correspondence to its exponents from level $n$ to level $i$) has a star as its last entry, and false otherwise.

**Algorithm 2** Procedure to generate the candidate list for the current maximal variable (subroutine).

1: procedure CANDIDATEVAR($M, C$) \[\triangleright M \text{ is a list of terms; } C \text{ is a list of variables.}\]
2: \[Y := C\]
3: \[\text{for } i = 1, \ldots, |C| \text{ do}\]
4: \[D_i := \{\beta \in \mathbb{N} | \exists t \in M, \deg_C(t) = \beta\}\]
5: \[\text{if for some } \gamma_1 \in D_i, \gamma_1 < \max(D_i), \gamma_1 + 1 \notin D_i, \text{ then}\]
6: \[\text{Delete } C[i] \text{ from } Y\]
7: \[\text{end if}\]
8: \[\text{end for}\]
9: \[\text{return } Y\]
10: end procedure

**Algorithm 3** Procedure to generate the candidate list for the current maximal variable.

1: procedure CANDIDATES($L, C$) \[\triangleright L \text{ is a list of lists of terms; } C \text{ is a list of variables.}\]
2: \[\text{for } i = 1, \ldots, |L| \text{ do}\]
3: \[Y[i] := \text{CANDIDATEVAR}(L[i], C);\]
4: \[\text{end for}\]
5: \[\text{return } \bigcap_i Y[i]\]
6: end procedure
Algorithm 4 Friends

1: procedure FRIENDS(A, Y, x_i, T) ⊢ T is the output of a previous execution of Friends (or it is empty), so it is formed by sets of the form T(t, x_j), t terms in the given set, and x_j variables.
2: for j = 1, . . . , μ(i) − 1 do
3:  
4:  
5: for t ∈ A^{(i)} do
6:  U(t, x_i) = ((u, V) | u ∈ B’ and V : tx_i = um, m ∈ T[V])
7:  if U(t, x_i) = ∅ then return ∅
8: end if
9: end for
10: if T ̸= ∅ then
11: for j = 1, . . . , μ(i) do
12:  
13:  
14: for t ∈ A^{(i)} do
15:  U(t, y) = ∅
16:  for (u, V) ∈ T(t, y) do
17:  if x_i ̸∈ V then
18:  U(t, y) = (u, V) ∪ U(t, y), U = U ∪ U(t, y)
19:  else
20:  if x_i ∈ V and Star(x_i, t) = true then
21:  U(t, y) = (u, V) ∪ U(t, y), U = U ∪ U(t, y)
22:  end if
23: end if
24: end for
25: if U(t, y) = ∅ then
26: return ∅
27: end if
28: end for
29: end for
30: end for
31: end if
32: return U
33: end procedure
Algorithm 5 Common

1: procedure COMMON(A, X, x_i, T) ⊢ T is the output of a previous execution of Friends (or it is empty), so it is formed by sets of the form T(t, x_j), t terms in the given set, and x_j variables.
2: Y = X \ {x_i}
3: Y′ = CANDIDATES(A, Y)
4: if |Y′| = 0 then return ∅
5: end if
6: for x_i ∈ Y′ do
7: for j = 1, . . . , μ(i) do
8: construct the l-bars C = {A(l) m} over A(i)
9: end for
10: U = Friends(C, Y, x_j, T)
11: if U = ∅ then continue
12: end if
13: if |A(l) m| = 1, ∀1 ≤ m ≤ μ(l) then ord = Y ∪ {x_i} return ord
14: end if
15: if Y ̸= ∅ then
16: C = COMMON(C, Y, x_i, U)
17: else
18: if Y = ∅ then ord = ord ∪ {x_i}
19: return ord
20: end if
21: end if
22: if ord ̸= ∅ then ord = ord ∪ {x_i}
23: else continue
24: end if
25: end for
26: return ∅
27: end procedure

References

1. Cartan, E.: Sur l’intégration des systèmes d’équations aux différentielles totales. Ann. Éc. Norm. 3e série 18, 241, (1901)
2. Cartan, E.: Sur la structure des groupes infinis de transformations. Ann. Éc. Norm. 3e série 21, 153, (1904)
3. Cartan, E.: Sur les systèmes en involution d’équations aux dérivées partielles du second ordre à une fonction inconnue de trois variables indépendentes. Bull. Soc. Marth. 39, 356 (1920)
4. Ceria, M.: A proof of the “Axis of Evil theorem” for distinct points. Rendiconti del Seminario Matematico dell’Università e del Politecnico di Torino 72(3–4), 213–233 (2014)
5. Ceria, M.: Bar Code for monomial ideals. J. Symbol. Comput. (2019). https://doi.org/10.1016/j.jsc.2018.06.012 J. Symbol. Comput., Vol 91: 30-56
6. Ceria, M.: Bar code: a visual representation for finite set of terms and its applications. Math. Comput. Sci. 14(2), 497-513 (2020)
7. Ceria, M.: Bar code versus janet tree, Atti dell’Accademia Peloritana dei Pericolanti, Vol 97, No 2 (2019)
8. Ceria, M.: Bar Code and Janet-like division, Atti dell’Accademia Peloritana dei Pericolanti, Classe di Scienze Fisiche, Matematiche e Naturali, Vol.100 N.1 https://doi.org/10.1478/AAPP.1001A2
9. Ceria, M., Mora, T.: Combinatorics of ideals of points: a Cerlienco-Mureddu-like approach for an iterative lex game, arXiv:1805.09165
10. Ceria, M., Mora, T., Roggero, M.: Term-ordering free involutive bases, https://doi.org/10.1016/j.jsc.2014.09.005. J. Symbol. Comput. (2015)
11. Ceria, M., Mora, T., Visconti, A.: Degroebnerization and its applications: a new approach for data modelling, preprint
12. Cerlienco, L., Mureddu, M.: Algoritmi combinatori per l’interpolazione polinomiale in dimensione ≥ 2, Publ. I.R.M.A. Strasbourg, 461/S-24 Actes 24° Séminaire Lotharingien, p.39-76 (1993)
13. Cerlienco, L., Mureddu, M.: From algebraic sets to monomial linear bases by means of combinatorial algorithms. Discrete Math. 139, 73–87 (1995)
14. Cerlienco, L., Mureddu, M.: Multivariate interpolation and standard bases for macaulay modules. J. Algebra 251, 686–726 (2002)
15. Delassus, E.: Extension du théorème de Cauchy aux systèmes les plus généraux d’équations aux dérivées partielles. Ann. Éc. Norm. 3e série 13, 421–467 (1896)
16. Delassus, E.: Sur les systèmes algébriques et leurs relations avec certains systèmes d’equations aux dérivées partielles. Ann. Éc. Norm. 3e série 14, 21–44 (1897)
17. Delassus, E.: Sur les invariants des systèmes différentiels. Ann. Éc. Norm. 3e série 25 255–318, (1908)
18. Eisenbud, D.: Commutative Algebra: with a View Toward Algebraic Geometry. Springer, USA (2013)
19. Felszeghy, B., Ráth, B., Rónyai, L.: The lex game and some applications. J. Symbolic Comput. 41, 663–681 (2006)
20. Galligo, A.: A propos du théorème de préparation de Weierstrass, L. N. Math.40, Springer, 543–579, (1974)
21. Gerdt, V.P., Blinkov, Y.A.: Involutive bases of Polynomial ideals. Math. Comp. Sim. 45, 543–560 (1998)
22. Gerdt, V.P., Blinkov, Y.A.: Minimal involutive bases. Math. Comp. Sim. 45, 519–541 (1998)
23. Gerdt, V.P., Blinkov, Y.A.: Involutive Division Generated by an Antigraded Monomial Ordering L. N. Comp. Sci 6885, Springer, 158-174, (2011)
24. Gerdt, V.P., Blinkov, Y.A.: Janet-like monomial division. International workshop on computer algebra in scientific computing, Springer, 174–183, (2005)
25. Gerdt, V.P., Blinkov, Y.A.: Janet-like Gröbner bases, international workshop on computer algebra in scientific computing, Springer, 184–195, (2005)
26. Gerdt, V., Blinkov, Y., Yanovich, D.: Construction of Janet bases I. Monomial bases, in Computer Algebra in Scientific Computing CASC, 233-247 (2001)
27. Grauert, H.: Über die Deformation isolierter Singularitäten analytischer Mengen. Inventiones mathematicae 15, 171-198, (1971/72)
28. Green, M., Stillman, M.: A tutorial on generic initial ideals. In: Buchberger, B., Winkler, F. (eds.) Grobner Bases Appl., pp. 90–108. Press, Cambridge Univ (1998)
29. Gunther, N.: Sur la forme canonique des systèmes déquations homogènes (in russian) [Journal de l’Institut des Ponts et Chaussées de Russie] Izdanie Inst. Puti, 1. 84, (1913)
30. Gunther, N.: Sur la forme canonique des equations algébriques C.R. Acad. Sci. Paris 157, 577–80 (1913)
31. Gunther, N.: Sur les modules des formes algébriques Trudy Tbilis. Mat. Inst. 9, 97–206 (1941)
32. Herzog, J.: A survey on stanley depth. In monomial ideals, computations and applications, Springer, Berlin, Heidelberg 3-45, (2013)
33. Hironaka, H.: Idealistic exponents of singularity In: Algebraic Geometry, The Johns Hopkins Centennial Lectures. 52-125 (1977)
34. Janet, M.: Sur les systèmes d’équations aux dérivées partielles, J. Math. Pure et Appl., 3, 65–151, (1920)
35. Janet, M.: Les modules de formes algébriques et la théorie générale des systemes différentiels, Annales scientifiques de l’École Normale Supérieure, (1924)
36. Janet, M.: Les systèmes d’équations aux dérivées partielles, Gauthier-Villars, (1927)
37. Janet, M.: Lecons sur les systèmes d’équations aux dérivées partielles , Gauthier-Villars (1929)
38. Mora, T.: Solving Polynomial Equation Systems. 4 Vols., Cambridge University Press, I (2003), II (2005), III (2015), IV (2016)
39. Mumford, D.: Lectures on Curves on an Algebraic Surface. Princeton Univ. Press (1966)
40. Pommaret, J. F.: Systems of partial differential equations and Lie pseudogroups, Gordon and Brach (1978)
41. Pommaret, J.F., Akli, H.: Effective methods for systems of algebraic partial differential equations, Progress in Mathematics 94, 411–426, Birkhäuser, (1990)
42. Riquier, C.: Les systèmes d’équations aux dérivées partielles, Gauthiers-Villars, (1910)
43. Robinson, L.B.: A new canonical form for systems of partial differential equations. Am. J. Math. 39, 95–112 (1917)
44. Robinson, L.B.: Sur les systèmes d’équations aux dérivées partialles C.R. Acad. Sci. Paris 157, 106–108, (1913)
45. Seiler, W.M.: Involution: The Formal Theory of Differential Equations and its Applications in Computer Algebra, Vol.24, Springer Science & Business Media, (2009)

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.