Simplicial Tree Computations

Extended Abstract

Massimo Caboara* Sara Faridi† Peter Selinger‡

May 29, 2005

Abstract

We present an algorithm that checks in polynomial time whether a simplicial complex is a tree. We also present an efficient algorithm for checking whether a simplicial complex is grafted. These properties have strong algebraic implications for their corresponding facet ideals.

1 Introduction

The main goal of this paper is to demonstrate that it is possible to check, in polynomial time, if a monomial ideal is the facet ideal of a simplicial tree.

Facet ideals were introduced in [6] (generalizing [11] and [10]) as a method to study square-free monomial ideals. The idea is to associate a simplicial complex to a square-free monomial ideal, where each facet (maximal face) of the complex is the collection of variables that appear in a monomial in the minimal generating set of the ideal (see Definition 2.4). The ideal will then be called the "facet ideal" of this simplicial complex. Special simplicial complexes are called "simplicial trees" (Definition 2.9). Facet ideals of trees have many properties; for example, they have normal and Cohen-Macaulay Rees rings [6]. Finding such classes of ideals is in general a very difficult problem. Simplicial trees also have very strong Cohen-Macaulay properties: their facet ideals are always sequentially Cohen-Macaulay [8], and one can determine under precisely what combinatorial conditions on the simplicial tree the facet ideal is Cohen-Macaulay [7]. In [9] it is shown that the theory is not restricted to square-free monomial ideals; via polarization, one can extend many properties of facet ideals to all monomial ideals. All these properties, and many others, make simplicial trees extremely useful from an algebraic point of view.

But how does one determine if a given square-free monomial ideal is the facet ideal of a simplicial tree? In Section 3, we show that this can be decided in polynomial time.

*Department of Mathematics, University of Pisa, caboara@dm.unipi.it.

†Department of Mathematics, University of Ottawa, faridi@uottawa.ca. Research supported by NSERC.

‡Department of Mathematics, University of Ottawa, selinger@mathstat.uottawa.ca. Research supported by NSERC.

This extended abstract is organized as follows: in Section 2 we introduce the notion of a simplicial complex, a facet complex, a tree, and a cycle. Section 3 contains the main theoretical result that enables us to produce a polynomial time algorithm to decide whether a given simplicial complex is a tree. The algorithm itself is introduced in Section 3.1, and the complexity and optimizations are discussed in Sections 3.2 and 3.3. Section 4 focuses on the algebraic properties of facet ideals: in Section 4.1 we discuss a method of adding generators to a square-free monomial ideal (or facets to the corresponding complex) so that the resulting facet ideal is Cohen-Macaulay. This method is called "grafting" a simplicial complex. For simplicial trees, being grafted and being Cohen-Macaulay are equivalent conditions [7]. We then introduce an algorithm that checks whether or not a given simplicial complex is grafted in Section 4.2, and discuss its complexity in Section 4.3.

Some proofs have been sketched or omitted in this extended abstract. Full proofs will appear in the forthcoming paper [3].

Implementations. The algorithms described in this paper have first been coded in CoCoAL, the program language of the CoCoA system [4]. These prototypical implementations can be downloaded from [2]. Much more efficient (but not so user friendly) C++ implementations are being developed using the cocoalib framework [5]. The C++ code will be available in the full paper according to the specifications of AJCA [1].

2 Simplicial complexes and trees

2.1 Definitions and notation

We define the basic notions related to facet ideals. More details and examples can be found in [6] and [7].

Definition 2.1 (Simplicial complex, facet). A *simplicial complex* Δ over a set of vertices $V = \{v_1, \ldots, v_n\}$ is a collection of subsets of V, with the property that $\{v_i\} \in \Delta$ for all i, and if $F \in \Delta$ then all subsets of F are also in Δ (including the empty set). An element of Δ is called a *face* of Δ, and the maximal faces are called *facets* of Δ.

Since we are usually only interested in the facets, rather than all faces, of a simplicial complex, it will be convenient to work with the following definition:

Definition 2.2 (Facet complex). A *facet complex* is a finite set Δ of finite sets, such that for all $F, G \in \Delta$, $F \not\subseteq G$. Each $F \in \Delta$ is called a *facet* of Δ, and each $v \in F$ is called a *vertex* of F and of Δ.

Remark 2.3 (Equivalence of simplicial complexes and facet complexes). The set of facets of a simplicial complex forms a facet complex. Conversely, the set of subsets of the facets of a facet complex is a simplicial complex. Therefore, every property of simplicial complexes corresponds to a property of facet complexes and vice versa. In this paper, we will work with facet complexes, but the results of course equally apply to simplicial complexes.

We define facet ideals as follows, giving a one-to-one correspondence between facet complexes (or, equivalently, simplicial complexes) and square-free monomial ideals.

Definition 2.4 (Facet ideal of a facet complex, facet complex of an ideal).

- Let Δ be a facet complex whose vertices are contained in $\{v_1, \dots, v_n\}$. Let k be a field, and let $R = k[x_1, \dots, x_n]$ be the polynomial ring with indeterminates $x_1, \dots, x_n$. The *facet ideal of* Δ is defined to be the ideal of R generated by all the square-free monomials $x_{i_1} \dots x_{i_s}$, where $\{v_{i_1}, \dots, v_{i_s}\}$ is a facet of Δ. We denote the facet ideal of Δ by $\mathcal{F}(\Delta)$.
- Let $I = (M_1, \dots, M_q)$ be an ideal in the polynomial ring $k[x_1, \dots, x_n]$, where k is a field and $M_1, \dots, M_q$ are square-free monomials in $x_1, \dots, x_n$ that form a minimal set of generators for I. The *facet complex of* I is defined to be $\delta_{\mathcal{F}}(I) = \{F_1, \dots, F_q\}$, where for each i, $F_i = \{v_j \mid x_j | M_i,\ 1 \leq j \leq n\}$.

From now on, we often use $x_1, \dots, x_n$ to denote both the vertices of Δ and the variables appearing in $\mathcal{F}(\Delta)$. We also sometimes ease the notation by denoting facets by their corresponding monomials; for example, we write xyz for the facet $\{x, y, z\}$.

We now generalize some notions from graph theory to facet complexes. Note that a graph can be regarded as a special kind of facet complex, namely one in which each facet has cardinality 2.

2.2 Simplicial trees

Simplicial trees are a generalization of graph-theoretic trees, in light of the fact that a graph can be regarded as a special kind of facet complex, where the facets are the edges of the graph.

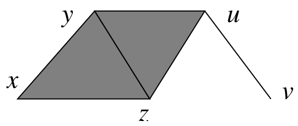
Definition 2.5 (Path, connected facet complex). Let Δ be a facet complex. A sequence of facets $F_1, \dots, F_n$ is called a *path* if for all $i = 1, \dots, n-1$, $F_i \cap F_{i+1} \neq \emptyset$. We say that two facets F and G are *connected* in Δ if there exists a path $F_1, \dots, F_n$ with $F_1 = F$ and $F_n = G$. Finally, we say that Δ is *connected* if every pair of facets is connected.

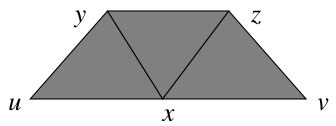
Notation 2.6. If F, G and H are facets of Δ, $H \leq_F G$ means that $H \cap F \subseteq G \cap F$, i.e. H is a subset of G "inside" F. The relation $\leq_F$ defines a preorder (reflexive and transitive relation) on the facet set of Δ.

Definition 2.7 (Leaf, joint). Let F be a facet of a facet complex Δ. Then F is called a *leaf* of Δ if either F is the only facet of Δ, or else there exists some $G \in \Delta \setminus \{F\}$ such that for all $H \in \Delta \setminus \{F\}$, we have $H \leq_F G$. If $F \cap G \neq \emptyset$, the facet G above is called a *joint* of the leaf F.

Example 2.8. In the facet complex $\Delta = \{xyz, yzu, uv\}$, xyz and uv are leaves, but yzu is not a leaf. Similarly, in $\Delta' = \{xyu, xyz, xzv\}$, the only leaves are xyu and

xzv.

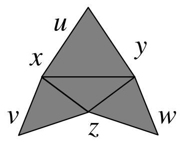
$\Delta =$ 

$\Delta' =$ 

Definition 2.9 (Forest, tree). A facet complex Δ is a *forest* if every nonempty subset of Δ has a leaf. A connected forest is called a *tree* (or sometimes a *simplicial tree* to distinguish it from a tree in the graph-theoretic sense).

When Δ is a graph, the notion of a simplicial tree coincides with that of a graph-theoretic tree.

Example 2.10. The facet complexes in Example 2.8 are trees. The facet complex $\Delta = \{xyu, xyz, xzv, yzw\}$ pictured below has three leaves xyu, xvz and yzw; however, it is not a tree, because if one removes the facet xyz, the remaining facet complex has no leaf.



2.3 Cycles

Definition 2.11 (Cycle). A nonempty facet complex is a *cycle* if it has no leaf. A cycle is *minimal* if none of its proper subsets are cycles.

Remark 2.12. Clearly a facet complex is a tree if and only if it does not have a subset which is a cycle.

The main tool used in the paper to prove Theorem 3.7 is the structure of cycles.

Definition 2.13 (Strong neighbor). Let Δ be a facet complex and $F, G \in \Delta$. We say that F and G are *strong neighbors*, written $F \sim G$, if for all $H \in \Delta$, $F \cap G \subseteq H$ implies $H = F$ or $H = G$.

The relation $\sim$ is symmetric, i.e. $F \sim G$ if and only if $G \sim F$, and reflexive, i.e. $F \sim F$.

It turns out that a cycle can be described as a sequence of strong neighbors. The following lemma follows directly from Definition 2.13.

Lemma 2.14. *If Δ is a facet complex with distinct facets F, G_1, G_2 such that $F \sim G_1$ and $F \sim G_2$, then F is not a leaf of Δ.*

Corollary 2.15. *Let Δ be a facet complex, and let $F_1, \dots, F_n$ be distinct facets with $n \geq 3$, such that for all $i = 1, \dots, n-1$, $F_i \sim F_{i+1}$ and $F_n \sim F_1$ in Δ. Then $\{F_1, \dots, F_n\}$ is a cycle.*

Lemma 2.16. *Suppose Δ is a minimal cycle, and let $n = |\Delta|$. Then $n \geq 3$, and the facets of Δ can be enumerated in such a way that $\Delta = \{F_1, \ldots, F_n\}$, and for all i, $F_i \sim F_{i+1}$, as well as $F_n \sim F_1$. Moreover, in all other cases, $F_i \not\sim F_j$ (so that each facet is a strong neighbor of precisely two other facets).*

The proof of this last lemma is based on the observation that every non-trivial tree has at least two leaves [6]. Further, if F is not a leaf of Δ, but is a leaf of $\Delta \setminus \{G\}$ for some facet G, then F and G must be strong neighbors in Δ.

3 Characterization of trees

As mentioned earlier, facet ideals of simplicial trees have strong algebraic properties. For example, they have normal and Cohen-Macaulay Rees rings. One would therefore like to be able to decide whether a given facet complex is a tree or not. We refer to this problem as the *decision problem for simplicial trees*.

Note that the naïve algorithm (namely, checking whether every non-empty subset has a leaf) is extremely inefficient: for a facet complex of n facets, there are $2^n - 1$ subsets to check. Also note that the definition of a tree is not inductive in any obvious way: for instance, attaching a single leaf to a tree need not yield a tree, as Example 2.10 shows. This seems to rule out an easy recursive algorithm.

We now demonstrate that the decision problem for simplicial trees can in fact be solved in polynomial time.

Definition 3.1. A *precomplex* is a finite multiset (a set in which repeated elements are allowed) Π of finite sets. Just like for facet complexes, an element $F \in \Pi$ is called a facet and an element $v \in F$ is called a vertex.

Remark 3.2. Trivially, any facet complex is also a precomplex. On the other hand, $\Pi = \{xyz, xyz, xz\}$ is an example of a precomplex which is not a facet complex. It fails to be a facet complex because (a) the facet xyz occurs more than once, and (b) the facet xz is a subset of the facet xyz.

Definition 3.3 (Reduction). Let $\Delta = \{F_1, \ldots, F_n\}$ be a facet complex, and let V be a set of vertices. We define the *reduction* of Δ along V to be the precomplex

$$\Delta \setminus\!\!\setminus V := \{F_1 \setminus V, \ldots, F_n \setminus V\}.$$

Note that in general, $\Delta \setminus\!\!\setminus V$ is not a facet complex.

Definition 3.4 (Residue). Let Δ be a facet complex, and let F, G_1, G_2 be three distinct facets. Let

$$\uparrow_F G_i := \{H \in \Delta \mid G_i \leq_F H \text{ and } H \neq G_i\ \}.$$

Let $S = \uparrow_F G_1 \cup \uparrow_F G_2$ and define the $\langle F, G_1, G_2\rangle$*-residue* of Δ to be the following precomplex:

$$\Delta_F^{G_1,G_2} = (\Delta \setminus S) \setminus\!\!\setminus F.$$

Remark 3.5. Note that in the expression $(\Delta \setminus S) \setminus\!\!\setminus F$, the set S is a set of facets, while F is a set of vertices. Also note that $F \in S$. Further, if $G_1 \not\leq_F G_2$ and $G_2 \not\leq_F G_1$, then $G_1, G_2 \notin S$.

Definition 3.6 (Triple condition). Let Δ be a facet complex. We say a triple of facets $\langle F, G_1, G_2 \rangle$ satisfies the *triple condition* if $G_1 \not\leq_F G_2$ and $G_2 \not\leq_F G_1$, and if $G_1 \setminus F$ and $G_2 \setminus F$ are connected in $\Delta_F^{G_1,G_2}$.

Our central claim is the following:

Theorem 3.7 (Main Theorem). *Let Δ be a facet complex. Then Δ is a tree if and only if no triple of facets in Δ satisfies the triple condition.*

Sketch of the proof. "$\Rightarrow$": Suppose there is a triple of facets $\langle F, G_1, G_2 \rangle$ satisfying the triple condition. Let $\{H_1, \ldots, H_n\}$ be a minimal path connecting $G_1 \setminus F$ and $G_2 \setminus F$ in $\Delta_F^{G_1,G_2}$. Choose $K_i \in \Delta$ such that $H_i = K_i \setminus F$ for all i, and such that $K_1 = G_1$ and $K_n = G_2$. Using Corollary 2.15, one can show that $\{F, K_1, \ldots, K_n\}$ is a cycle in Δ.

"$\Leftarrow$": Suppose that Δ is not a tree. Then Δ has some minimal cycle, which can be written as $\{F_1, \ldots, F_n\}$ satisfying the condition of Lemma 2.16. Then it can be shown that the triple $\langle F_1, F_2, F_n \rangle$ satisfies the triple condition. $\square$

Corollary 3.8. *If F is part of a minimal cycle, then there exist some $G_1, G_2 \in \Delta$ such that $\langle F, G_1, G_2 \rangle$ satisfies the triple condition.* $\square$

3.1 A polynomial-time tree decision algorithm

By Theorem 3.7, to check if a facet complex $\Delta = \{G_1, \ldots, G_l\}$ is a tree, we only need to check the triple condition for all triples of elements of Δ. The checks themselves are straightforward. Since the triple condition for $\langle F, G, G' \rangle$ is clearly unchanged if one switches G and G', we can limit triple checking to the elements of the set $\{\langle F, G_i, G_j \rangle \in \Delta^3 \mid G_i \neq F \neq G_j, i < j\}$. The procedures for the basic steps follow immediately from the earlier definitions.

Algorithm 3.9 (Tree decision algorithm).
Input: a facet complex $\Delta = \{G_1, \ldots, G_l\}$ with n vertices.
Output: **True** if Δ is a tree, **False** otherwise.

1. For each triple $\langle F, G, G' \rangle \in \{\langle F, G_i, G_j \rangle \in \Delta^3 \mid G_i \neq F \neq G_j, i < j\}$
   (a) If $G \leq_F G'$ or $G' \leq_F G$, continue with the next triple.
   (b) Let $\Pi = \Delta_F^{G,G'}$.
   (c) If $G \setminus F$ and $G' \setminus F$ are connected in Π, return **False**.
2. Return **True**.

The algorithm uses very little memory; Δ and Π require nl bits each, and the memory required to perform the connectedness check and to store the various counters is negligible. Thus, the total memory usage is roughly twice the amount necessary for Δ; memory locality is hence quite good, and computation can generally take place in the cache. We will hence only deal with time complexity.

3.2 Complexity

In the worst case we have to check $3 \cdot \binom{l}{3} = l(l-1)(l-2)/2$ triples. For each triple, the cost of step (a) is $O(n)$, the cost of step (b) is $O(nl)$ and the cost of step (c) is $O(nl)$. The total time complexity of the algorithm is therefore $O(nl^4)$.

Example 3.10. Consider the facet complex $\Delta = \{xy, xz, yz, yu, zt\}$. We have to check $3 \cdot \binom{5}{3} = 30$ triples. We start with the triple $\langle xy, xz, yz \rangle$.

- $xz \not\leq_{xy} yz$ since $xy \cap xz = x \not\subseteq y = xy \cap yz$. Similarly $yz \not\leq_{xy} xz$.
- $xz \setminus xy = z$ and $yz \setminus xy = z$ are connected (they are equal) in the precomplex $\Delta_{xy}^{xz,yz} = \{z, z, u, zt\}$.

We have hence discovered that Δ is not a tree. A more unlucky choice of facets could have brought about the checking of 27 useless triples before the discovery that Δ is not a tree, the other two useful triples being $\langle yz, xy, xz \rangle$ and $\langle xz, xy, yz \rangle$.

Example 3.11. Some statistics for a bigger random example. Consider the facet complex $\Delta = \{lka$, qik, $tykj$, wuv, rjb, $eioab$, gdc, zv, rtj, $qrvm$, gzm, $tgzb$, $rgvm$, $qlav$, $qeocn$, $ikfaz$, bn, $ekjs$, $pfvn$, $wtodv\}$. We discover that it is not a tree after checking 4 facets; we performed the connectedness check only once. If one checks all $3 \cdot \binom{20}{3} = 3420$ triples, one finds that 445 of them require a connectedness check, and 418 of them reveal that Δ is not a tree.

Example 3.12. The facet complex $\{x_i x_{i+1} x_{i+2} \mid i = 1, \ldots, 400\}$ is trivially a tree. Checking this by a direct application of Algorithm 3.9 requires dealing with $3 \cdot \binom{400}{3} = 31,760,400$ triples, and takes about 24.6 seconds on an Athlon 2600+ machine for our C++ implementation. All the timings in the remainder of this paper refer to this machine.

3.3 Optimization

The runtime of Algorithm 3.9 can be improved by introducing some optimizations. First, note that if F is a facet such that no triple $\langle F, G, G' \rangle$ satisfies the triple condition, then by Corollary 3.8, F cannot be part of any minimal cycle of Δ. Therefore, F can be removed from Δ, reducing the number of subsequent triple checks. We refer to this optimization as the *removal of useless facets*.

Example 3.13. We check the tree $\{x_i x_{i+1} x_{i+2} \mid i = 1, \ldots, 400\}$ of Example 3.12 with a version of Algorithm 3.9 with removal of useless facets. This requires dealing with $10,586,800$ triples and takes about 5.6 seconds.

An important special case of a "useless facet" is a reducible leaf, as captured in the following definition:

Definition 3.14 (Reducible leaf). A facet F of a facet complex Δ is called a *reducible leaf* if for all $G, G' \in \Delta$, either $G \leq_F G'$ or $G' \leq_F G$.

A reducible leaf is called a "good leaf" by Zheng [12].

Remark 3.15. The facet F is a reducible leaf of Δ if and only if F is a leaf of every $\Delta' \subseteq \Delta$ with $F \in \Delta'$.

The remark immediately implies that a reducible leaf cannot be part of a cycle. Thus, it can be removed from Δ, and the algorithm can then be recursively applied to $\Delta' = \Delta \setminus \{F\}$. In our experience, most simplicial trees possess a reducible leaf; in fact, Zheng [12] conjectured that this is always the case. Checking whether a given facet F is a reducible leaf requires ordering all facets with respect to $\leq_F$, which takes $O(nl \log l)$ steps. A reducible leaf can thus be found in time $O(nl^2 \log l)$. Therefore, if Zheng's conjecture is true, the tree problem can be decided in time $O(nl^3 \log l)$. But even if the conjecture is not true, removing all reducible leaves at the beginning of Algorithm 3.9 is still a worthwhile optimization.

Sparse Complexes

Let Δ be a facet complex with l facets. If every $F \in \Delta$ intersects a substantial ($\sim l$) number of facets, then the number of cycles is probably high and our algorithm is usually able to detect one of them easily. If this does not happen, we can exploit the facet complex "sparseness" in our algorithm.

Definition 3.16 (Connectivity bound, sparse complex). Let Δ be a facet complex with l facets. We say that Δ has *connectivity bound* d if every facet intersects at most d other facets. Moreover, if $d \ll l$ then Δ is called a *sparse complex*.

Let us suppose that the facet complex Δ is over n vertices, has l facets, and its connectivity bound is d. To check if Δ is a tree it is sufficient to check the connected triples only. For each facet F (l facets): first construct the set of all facets G connected to F (called the *connection set*, at cost $O(nl)$), then for all G, G' in the set (d^2 pairs) perform the triple check on $\langle F, G, G' \rangle$ (cost $O(nl)$ per triple). The total cost is $O(nl^2d^2)$. The space required to construct the connection sets is $O(d)$, hence negligible. For sparse examples, this optimization is clearly worthwhile:

Example 3.17. We check the tree $\{x_i x_{i+1} x_{i+2} \mid i = 1, \ldots, 400\}$ of Example 3.12 with the algorithm detailed above. We deal with 398 triples and spend 0.2 seconds.

Example 3.18. The facet complex $\{x_i x_{i+1} \cdots x_{i+200} \mid i = 1, \ldots, 3200\}$ is a tree. Tree checking with the algorithm detailed above requires dealing with $61,013,400$ triples, and takes about 190 seconds. Without any optimization, the number of triples to check is $16,368,643,200$ and the time spent by the algorithm is > 2 days.

4 Algebraic properties of facet ideals

We now study facet ideals from a more algebraic point of view. In particular, we are interested in ways to determine whether a given facet complex Δ is Cohen-Macaulay, meaning whether $R/\mathcal{F}(\Delta)$ is a Cohen-Macaulay ring. We first need to introduce some new terminology.

Definition 4.1 (Vertex covering number, unmixed facet complex, independence number). Let Δ be a facet complex.

- A *vertex cover* for Δ is a set A of vertices of Δ, such that $A \cap F \neq \emptyset$ for every facet F. The smallest cardinality of a vertex cover of Δ is called the *vertex covering number* of Δ and is denoted by $\alpha(\Delta)$. A vertex cover A is *minimal* if no proper subset of A is a vertex cover. A facet complex Δ is *unmixed* if all of its minimal vertex covers have the same cardinality.
- A set B of facets of Δ is called an *independent set* if $F \cap G = \emptyset$ for all $F, G \in B$. The maximum possible cardinality of an independent set of facets, denoted by $\beta(\Delta)$, is called the *independence number* of Δ.

Example 4.2. Consider the two facet complexes in Example 2.8. We have $\alpha(\Delta) = \beta(\Delta) = 2$. Also, Δ is unmixed as its minimal vertex covers $\{x, u\}$, $\{y, u\}$, $\{y, v\}$, $\{z, u\}$ and $\{z, v\}$ all have cardinality equal to two. We further have $\alpha(\Delta') = \beta(\Delta') = 1$, but Δ' is not unmixed, because $\{x\}$ and $\{y, z\}$ are minimal vertex covers of different cardinalities.

The following observations are basic but useful.

Proposition 4.3 (Cohen-Macaulay facet complexes [6, 7]). *Let Δ be a facet complex with vertices in $x_1, \ldots, x_n$, and consider its facet ideal $I = \mathcal{F}(\Delta)$ in the polynomial ring $R = k[x_1, \ldots, x_n]$. Then the following hold:*

(a) $\mathrm{height}\ I = \alpha(\Delta)$ *and* $\dim R/I = n - \alpha(\Delta)$.

(b) An ideal $p = (x_{i_1}, \ldots, x_{i_s})$ of R is a minimal prime of I if and only if the set $\{x_{i_1}, \ldots, x_{i_s}\}$ is a minimal vertex cover for Δ.

(c) If $k[x_1, \ldots, x_n]/\mathcal{F}(\Delta)$ is Cohen-Macaulay, then Δ is unmixed.

4.1 Grafting

One of the most basic ways to build a Cohen-Macaulay facet complex is via grafting.

Definition 4.4 (Grafting [7]). A facet complex Δ is a *grafting* of the facet complex $\Delta' = \{G_1, \ldots, G_s\}$ with the facets $F_1, \ldots, F_r$ (or we say that Δ is *grafted*) if

$$\Delta = \{F_1, \ldots, F_r\} \cup \{G_1, \ldots, G_s\}$$

with the following properties:

(i) $G_1 \cup \ldots \cup G_s \subseteq F_1 \cup \ldots \cup F_r$;

(ii) $F_1, \ldots, F_r$ are all the leaves of Δ;

(iii) $\{G_1, \ldots, G_s\} \cap \{F_1, \ldots, F_r\} = \emptyset$;

(iv) For $i \neq j$, $F_i \cap F_j = \emptyset$;

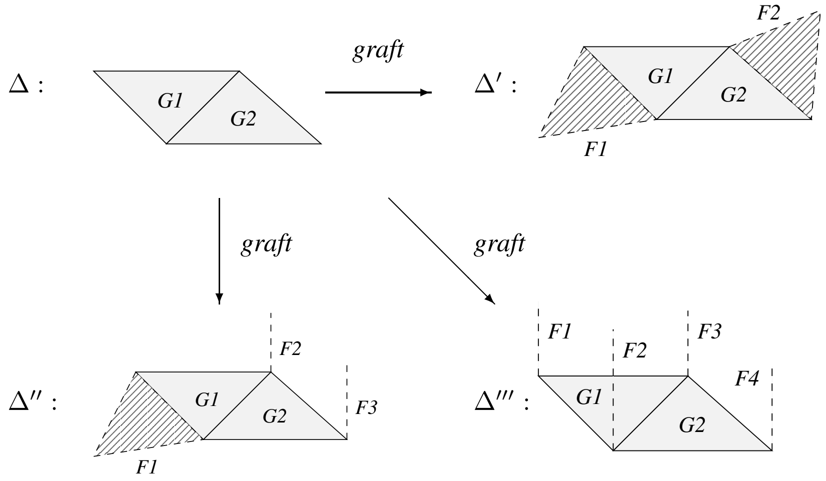


Figure 1: Three different ways of grafting a facet complex Δ.

(v) If G_i is a joint of Δ, then $\Delta \setminus \{G_i\}$ is also grafted.

Note that the definition is recursive, since graftedness of Δ is defined in terms of graftedness of $\Delta \setminus \{G_i\}$. Also note that a facet complex that consists of only one facet or several pairwise disjoint facets is grafted, as it can be considered as a grafting of the empty facet complex. It is easy to check that conditions (i) to (v) above are satisfied in this case. It is also clear that the union of two or more grafted facet complexes is itself grafted.

Example 4.5. There may be more than one way to graft a given facet complex. For example, some possible ways of grafting $\{G_1, G_2\}$ are shown in Figure 1.

The interest in grafted facet complexes, from an algebraic point of view, lies in the following facts.

Theorem 4.6 (Grafted facet complexes are Cohen-Macaulay). *Let Δ be a grafted facet complex. Then $\mathcal{F}(\Delta)$ is Cohen-Macaulay.*

Even more holds when Δ is a tree.

Theorem 4.7 ([7, Corollaries 7.8, 8.3]). *If Δ is a simplicial tree, then the following are equivalent:*

(i) Δ *is unmixed;*

(ii) Δ *is grafted;*

(iii) $\mathcal{F}(\Delta)$ *is Cohen-Macaulay.*

4.2 Graftedness algorithm

A direct application of Definition 4.4 is not very convenient for checking whether a given facet complex Δ is grafted, since at each step of the recursion, one potentially needs to check condition (v) for several of the G_i, and this leads to a worst-case exponential algorithm. In order to arrive at a more efficient algorithm, we characterize graftedness as follows:

Lemma 4.8 (cf. [7, Remark 7.2, 7.3]). *A facet complex Δ is grafted if and only if (1) for each vertex v, there exists a unique leaf F such that $v \in F$, and (2) all leaves of Δ are reducible.*

Sketch of the proof. First, assume that Δ is grafted. Condition (1) follows from (i), (ii) and (iv). The fact that all leaves are reducible is shown by induction on the number of facets of Δ. The converse is also shown by induction. Suppose Δ satisfies (1) and (2), and let $\{F_1, \ldots, F_r\}$ and $\{G_1, \ldots, G_s\}$ be the sets of leaves and non-leaves, respectively. Conditions (i)–(iv) hold trivially. Further, if G_i is a joint, then $F_1, \ldots, F_r$ are still reducible leaves of $\Delta \setminus \{G_i\}$ by Remark 3.15; also, there are no additional leaves in $\Delta \setminus \{G_i\}$. Therefore, $\Delta \setminus \{G_i\}$ satisfies (1) and (2) and is therefore grafted by induction hypothesis, proving (v). ☐

The algorithm for checking if a facet complex is grafted follows immediately from Lemma 4.8.

Algorithm 4.9 (Graftedness algorithm).
Input: A facet complex Δ with l facets and n vertices.
Output: **True** if Δ is grafted, **False** otherwise.

1. Build the lists $\mathcal{F} = \langle F_1, \ldots, F_k \rangle$ (leaves of Δ) and $\mathcal{G} = \langle G_1, \ldots, G_m \rangle$ (facets of Δ which are not leaves).
2. If $\bigcup_{G \in \mathcal{G}} G \not\subseteq \bigcup_{F \in \mathcal{F}} F$, return **False**.
3. If $\exists\, F, F' \in \mathcal{F}$ such that $F \cap F' \neq \emptyset$, return **False**.
4. If $\exists\, F \in \mathcal{F}$ that is not a reducible leaf, return **False**.
5. Return **True**.

4.3 Complexity

The leaf checking cost is $O(nl)$, hence the cost of step 1 is $O(nl^2)$. The cost of steps 2 and 3 is $O(nl)$. For step 4, there are k facets F to check. Checking whether F is reducible takes $O(nl \log l)$ steps as mentioned in Section 3.3. Therefore the total cost for step 4 is $O(nl^2 \log l)$, and this is the cost of the algorithm.

Example 4.10. Let $\Delta = \{xyz, yzu, ztu, uv, tw\}$, with $\mathcal{F} = \{xyz, uv, tw\}$ and $\mathcal{G} = \{yzu, ztu\}$. Then $\bigcup_{G \in \mathcal{G}} G \subseteq \bigcup_{F \in \mathcal{F}} F = \{x, y, z, t, u, v, w\}$ and $xyz \cap uv = xyz \cap tw = uv \cap tw = \emptyset$. Additionally, we check that each $F \in \mathcal{F}$ is a reducible leaf by showing that the set $\{F \cap G \mid G \in \mathcal{G}\}$ is a totally ordered set under inclusion. For example, if $F = xyz$, then this set is equal to $\{yz, z\}$ which is totally ordered. This holds for all $F \in \mathcal{F}$, and hence the facet complex is grafted.

Further work

As Theorem 3.7 and Algorithm 3.9 suggest, to check whether or not a given facet complex is a tree, it is not necessary to check if every subset is a cycle. On the other hand, it might be useful to have more information on the cycles of a facet complex.

The main ingredient in the proof of Theorem 4.7 is a generalization of König's theorem from graph theory.

Theorem 4.11 ([7] Theorem 5.3). *If Δ is a simplicial tree (forest) and $\alpha(\Delta) = r$, then Δ has r independent facets, and therefore $\alpha(\Delta) = \beta(\Delta) = r$.*

If Δ is a bipartite graph (not necessarily a tree), then the statement of Theorem 4.11 still holds. Moreover, facet ideals of bipartite graphs have and Cohen-Macaulay Rees rings [10]. These facts lead us to consider the question: "Is there a higher-dimensional generalization of a bipartite graph?"

The most promising approach so far has been to consider a facet complex "multipartite" if it has no minimal cycles of odd length. Computational evidence has shown that Theorem 4.11 probably holds for such facet complexes. Using Corollaries 2.15 and 3.8 and Lemma 2.16, we have developed an algorithm that detects minimal cycles in a given facet complex. Details of this work will be given in the full paper.

References

[1] C. Traverso, *The Active Journal for Computational Algebra: A Proposal*, available at `http://www.dm.unipi.it/~caboara/Research/SimplicialTrees/ajca.pdf`

[2] M. Caboara, S. Faridi, P. Selinger, Prototype implementation of tree algorithms, available at `http://www.dm.unipi.it/~caboara/Research/SimplicialTrees/Trees.coc`.

[3] M. Caboara, S. Faridi, P. Selinger, *Simplicial tree computations*, in preparation.

[4] CoCoA Team, *CoCoA: a system for doing Computations in Commutative Algebra*, available at `http://cocoa.dima.unige.it/`.

[5] CoCoALib. See `http://cocoa.dima.unige.it/cocoalib/`.

[6] S. Faridi, *The facet ideal of a simplicial complex*, Manuscripta Mathematica 109 (2002), 159–174.

[7] S. Faridi, *Cohen-Macaulay properties of square-free monomial ideals*, Journal of Combinatorial Theory, Series A, to appear.

[8] S. Faridi, *Simplicial trees are sequentially Cohen-Macaulay*, J. Pure and Applied Algebra 190 (2004), 121–136.

[9] S. Faridi, *Monomial ideals via square-free monomial ideals*, Lecture Notes in Pure and Applied Mathematics, to appear.

[10] A. Simis, W. Vasconcelos, R. Villarreal, *On the ideal theory of graphs*, J. Algebra 167 (1994), no. 2, 389–416.

[11] R. Villarreal, *Cohen-Macaulay graphs*, Manuscripta Math. 66 (1990), no. 3, 277–293.

[12] X. Zheng, *Homological properties of monomial ideals associated to quasi-trees and lattices*, Ph.D. thesis, Universität Duisburg-Essen, August 2004.