



New York University
A private university in the public service

Courant Institute of Mathematical Sciences

251 Mercer Street
New York, NY 10012

Telephone: (212) 998-3100

Fax: (212) 995-4121

FAX TRANSMISSION LEAD PAGE

From CIMS FAX No. (212) 995-4121

Date: 1/30/97

of pages transmitted
(including lead page): 13

FAX SENT TO:

FAX #: (510) 642 - 8609
Addressee: Jesús De Loera
Addressee's Phone #: -
Organization: -
Location: -

FROM:

Originator's Name: RICHARD POLLACK
Phone #: (212) 998 - 3167
Email address: pollack@geometry.nyu.edu

Sender's N

More reverse search - japanese
ideas on animating
translations.

(89)

Enumerating Triangulations for Arbitrary Configurations of Points and for Products of Two Simplices

ask Jesus

Fumihiko Takeuchi
fumi@is.s.u-tokyo.ac.jp

Hiroshi Imai
imai@is.s.u-tokyo.ac.jp

Department of Information Science, University of Tokyo, Tokyo, Japan 113

Abstract

We propose three algorithms to enumerate triangulations for arbitrary configurations of points and for products of two simplices. Since this product is highly symmetric, counting all triangulations naively is inefficient. We may be counting the "same" triangulation many times. The classes of triangulations with respect to symmetry should be counted. Our first algorithm enumerates these classes of regular triangulations, a subset of triangulation. Nonregular triangulations are still unexplored, and there has been known no efficient algorithms to enumerate all triangulations including non-regular ones. The remaining two algorithms we propose enumerate all triangulations for an arbitrary configuration of points. Our second algorithm regards triangulations as maximal independent sets of the intersection graph, and applies a general algorithm for such enumeration. An intersection graph is a graph with all maximal dimensional simplices the vertices and edges between those intersecting improperly. Our third algorithm regards triangulations as independent sets with connected internals. We enumerate such sets incrementally: recursively adding an adjacent simplex. This can be applied for the classes of (all) triangulations for the product. All of these three algorithms work in time proportional to the output. Furthermore they all require small memory: only for the size of several triangulations.

1 Introduction

Gel'fand, Kapranov and Zelevinsky introduced the secondary polytope for point configurations. In this polytope, the vertices correspond to regular triangulations of the point configuration, and those triangulations which can be modified by flips are connected by edges [7], [8]. Billera, Filliman and Sturmfels analyzed the complexity of computing this secondary polytope [3].

Regular triangulations of the product of two simplices $\Delta_k \times \Delta_l$, where k and l are their dimensions, have relations with other branches of mathematics, such as Gröbner bases [17]. This polytope is highly symmetric: it has the symmetry of the direct product of two symmetric groups $S_{k+1} \times S_{l+1}$. So, it is not smart to count all of them naively, because we may have counted the "same" thing $(k+1)!(l+1)!$ times. De Loera devised a program to enumerate regular triangulations for given sets of points. The program can take this symmetry into account, and he enumerated the triangulations, all of which are regular, for the case of $\Delta_2 \times \Delta_3$ and $\Delta_2 \times \Delta_4$ [4], [5]. When the dimensions become larger, even the number of classes divided by symmetry becomes huge. De Loera is using breadth first search in his program, so all visited triangulations should be kept in the memory, and the memory constraint becomes serious in larger cases.

Masada, Imai and Imai proposed an algorithm to enumerate regular triangulations with output-size sensitive time complexity, which is same as de Loera's, using the memory only of the size for two triangulations [13], [14]. It uses a general technique for enumeration which is called reverse search, by Avis and Fukuda [1], [2].

Our first algorithm in this paper enumerates the classes of regular triangulations for the product of two simplices with respect to symmetry. This polytope is highly symmetric, as mentioned above, so it is important to enumerate the classes. The algorithm runs in output-sensitive time, i.e. in proportional to the number of classes, and requires memory only for the size linear to a triangulation.

The second and third algorithm in this paper enumerates all triangulation, regular or not, of an arbitrary configuration of points. Though there are some results, the enumeration of all triangulations in dimension higher than two still remains to be unexplored [6]. In the second algorithm we regard triangulations as a subclass of maximal independent sets of the intersection graph of the maximal dimensional simplices, and apply an algorithm for enumerating maximal independent sets. This also runs in output-sensitive time, with the memory of the size for one triangulation. ← Wool

The third algorithm enumerates the independent sets whose internal is connected, in which all triangulations are included. In this algorithm we construct such independent sets incrementally by adding adjacent simplices to an existing set. This algorithm also uses reverse search, so the time complexity is output-sensitive and required memory is only of the size for two triangulations. A version for enumerating the classes of triangulations is also shown.

We begin by brief explanations of the concepts we use: regular triangulations and secondary polytopes (Section 2) and reverse search (Section 3). Next we derive some properties of products of two simplices (Section 4). Then we present our first result: enumeration of regular triangulations (Section 5). We prepare some notations for

enumerating all triangulations (Section 6). Finally we study this case by maximal independent sets (Section 7), and by connected independent sets (Section 8).

2 Regular triangulations and the secondary polytope

Regular triangulations are a certain kind of triangulations. They correspond to the vertices of a polytope, secondary polytope, which is determined uniquely by a configuration of points. Thanks to this property we can enumerate all regular triangulations applying a vertex enumeration method to the secondary polytope. Please refer to [3], [7], [8], [12] and [19] for discussions in detail.

Let $\mathcal{A} = \{a_1, \dots, a_n\} \subset \mathbb{R}^{k-1}$ be a configuration of points, their convex hull $Q = \text{conv}(\mathcal{A})$, with $\dim(Q) = k - 1$. A subset $\sigma \subset \mathcal{A}$ is a d dimensional *simplex*, or a d -*simplex* in short, if the points in σ are affinely independent and $\#\sigma = d + 1$.

Definition 2.1 (triangulation)

A simplicial complex T is a *triangulation* of (Q, \mathcal{A}) if its skelton $|T| = \cup T$ equals Q and its points are among \mathcal{A} .

In other words, triangulation T of (Q, \mathcal{A}) is a collection of $(k - 1)$ -simplices and their faces, with the maximal dimensional simplices having their vertices in \mathcal{A} and intersecting at their possibly empty faces.

Definition 2.2 (regular triangulation)

A triangulation T of (Q, \mathcal{A}) is *regular* if there exists a vector $\psi : \mathcal{A} \rightarrow \mathbb{R}$ having the following property. For $P = \text{conv}\{(a_1, \psi_1), \dots, (a_n, \psi_n)\}$, and π the projection

$$\pi : \mathbb{R}^k \rightarrow \mathbb{R}^{k-1} \quad \begin{pmatrix} x \\ x_k \end{pmatrix} \mapsto x,$$

$$T = \{\pi(F) : F \text{ is a lower face of } P\}.$$

Here F being a lower face means,

$$F = \{x \in P : cx = c_0\}, \quad cx \leq c_0 \text{ valid for } P, \quad c_{d+1} < 0.$$

For any triangulation, regular or not, a vector called volume vector corresponds. In this vector, each entry corresponds to a volume sum of the simplices having a point in the configuration \mathcal{A} as an vertex.

Definition 2.3 (volume vector)

Let T be a triangulation of (Q, \mathcal{A}) . The *volume vector* for T is a vector $\varphi_T : \mathcal{A} \rightarrow \mathbb{R}$ with

$$\varphi_T(\omega) = \sum_{\sigma \in T: \omega \in \text{vert}(\sigma)} \text{vol}(\sigma)$$

where vol is the volume function, and $\text{vert}(\sigma)$ the set of vertices of σ .

We map all triangulations T in a $(\#\mathcal{A})$ -dimensional space by regarding their volume vectors φ_T as coordinates.

Definition 2.4 (secondary polytope)

The *secondary polytope* $\Sigma(\mathcal{A})$ of a point configuration \mathcal{A} is the convex hull of φ_T in $\mathbb{R}^{\mathcal{A}}$, for all triangulations T of (Q, \mathcal{A}) .

Now we state that regular triangulations correspond to the vertices of the secondary polytope $\Sigma(\mathcal{A})$. The nonregular ones are mapped to the points other than the vertices, and their injectivity is not necessarily guaranteed.

Theorem 2.5 ([7, Chapter 7. Theorem 1.7.])

The secondary polytope $\Sigma(\mathcal{A})$ has dimension $n - k$, and its vertices correspond one-to-one to the points φ_T of regular triangulations of (Q, \mathcal{A}) .

The vertices connected by an edge in the secondary polytope are "similar". Indeed, they can be modified each other by "flips". In introducing flips, we use the idea of circuits.

A collection Z of points in \mathbb{R}^{k-1} is called a *circuit* if they are affinely dependent, and any of their proper subsets are independent. For a circuit Z , there exists a vector $c \in \mathbb{R}^Z$ unique up to affine multiplication, such that

$$\sum_{\omega \in Z} c_\omega \omega = 0, \quad \sum c_\omega = 0$$

We can decompose the set Z into Z_+ and Z_- according to the sign of c_ω , up to the whole exchange. The dimension of $\text{conv}(Z)$ can be smaller than $k - 1$.

Proposition 2.6 ([7, Chapter 7. Proposition 1.2.])

For any circuit $Z \subset \mathbb{R}^{k-1}$, only two triangulations T_+ and T_- exist for $(\text{conv}(Z), Z)$. Here the maximal dimensional simplices for T_+ are $\text{conv}(Z \setminus \{\omega\})$ for $\omega \in Z_+$, and T_- are $\text{conv}(Z \setminus \{\omega\})$ for $\omega \in Z_-$.

The two triangulations of a circuit suggest the flip modification. This modification can be applied to a triangulation only when this circuit "appears".

Definition 2.7

Let T be a triangulation of (Q, \mathcal{A}) and $Z \subset \mathcal{A}$ a circuit. T is supported on Z if the following holds.

- (1) There are no points of T in $\text{conv}(Z)$ other than Z itself.
- (2) $\text{conv}(Z)$ is a union of simplices in T .
- (3) For two maximal dimensional simplices $\text{conv}(I)$ and $\text{conv}(I')$ in the same triangulation of Z , and $F \subset \mathcal{A} \setminus Z$, a simplex $\text{conv}(I \cup F)$ appears in T if and only if $\text{conv}(I' \cup F)$ appears.

For a triangulation T supported on a circuit Z , the flip or modification along Z is a triangulation $s_Z(T)$ formed by simplices $\text{conv}(I' \cup F)$ in place of $\text{conv}(I \cup F) \in T$, where I and I' are in different triangulations of Z and $F \subset \mathcal{A} \setminus Z$, and the remaining simplices of T not of the form $\text{conv}(I \cup F)$. Clearly $s_Z(s_Z(T)) = T$. Even if T is a regular triangulation, $s_Z(T)$ is not necessarily regular. The subsequent theorem describes the relation between adjacency on $\Sigma(\mathcal{A})$ and this flip modification.

Theorem 2.8 ([7, Chapter 7. Theorem 2.10.])

Let T, T' be regular triangulations of (Q, \mathcal{A}) . Their corresponding vertices $\varphi_T, \varphi_{T'}$ are joined by an edge in $\Sigma(\mathcal{A})$ if and only if there exists a circuit $Z \subset \mathcal{A}$ supporting T and T' , and they are obtained from each other by modification along Z .

3 Reverse search

Reverse search is a general technique for enumeration. It performs at the same output-size sensitive time as breadth first search (BFS) or depth first search (DFS), but requires memory only for twice the size of an object among the whole we want to enumerate. BFS and DFS needs output-size sensitive memory size, because we have to memorize all reached vertices. In addition to the adjacency relation, which is necessary for BFS and DFS, parent-children relation is needful to execute reverse search. By this parent-children relation, we construct in implicit a spanning tree in the adjacency graph of the objects of enumeration, and perform a DFS on the tree. The word "implicit" means that we do not really compute the whole tree at once in the algorithm, but we do traverse it using local information [1], [2].

First we state the adjacency and parent-children relation for reverse search. This structure for reverse search is named "local search structure given by an A -oracle." We call it a structure for reverse search in this paper.

Definition 3.1

$(S, \delta, \text{Adj}, f)$ is a local search given by an A -oracle if it suffices the followings.

- S is a finite set.
- $\delta \in \mathbb{N}$
- $\text{Adj} : S \times \{1, \dots, \delta\} \rightarrow S \cup \{\emptyset\}$. For any $a \in S$ and $i, j \in \{1, \dots, \delta\}$,
 - $\text{Adj}(a, i) \neq a$
 - if $\text{Adj}(a, i) = \text{Adj}(a, j) \neq \emptyset$ then $i = j$.
- $f : S \rightarrow S$ is the parent function:

$$f(a) = a \text{ or } \text{Adj}(a, i) \text{ for some } i.$$

Furthermore, there exists a unique root vertex $r \in S$ such that $f(r) = r$. For any other vertex $a \neq r$, there exists $n \in \mathbb{N}$ such that $f^{(n)}(a) = r$.

S is the set to be enumerated. We set the maximum degree of the adjacency graph to δ . For each vertex $a \in S$ the adjacency function Adj returns its indexed adjacent vertex, or sometimes \emptyset if it has degree less than δ . This index is for use in the enumeration algorithm. An example of this structure is shown in Figure 1. The information of δ, Adj, f and r is given to the reverse search algorithm, and it returns S as its output. Actually we do not need r , because we can find it by applying f several times to a vertex. The algorithm is presented in Figure 2.

Theorem 3.2 ([2, Corollary 2.3.])

The algorithm in Figure 2 works for the structure in Definition 3.1. The time complexity is $O(\delta(\text{time}(\text{Adj}) + \text{time}(f)) \# S)$, where $\text{time}(\text{Adj})$ and $\text{time}(f)$ are the time necessary to compute functions Adj and f . The memory required is the size of two objects in S .

4 $\Delta_k \times \Delta_l$ and its symmetry

The standard d -simplex Δ_d is the convex hull $\text{conv}\{e_1, \dots, e_{d+1}\}$ in \mathbb{R}^{d+1} . We use e_i or f_j for unit vectors whose i -th or j -th element is one and zeros in the rest. The product of two standard simplices $\Delta_k \times \Delta_l$ is

$$\Delta_k \times \Delta_l = \text{conv}\left\{\begin{pmatrix} e_i \\ f_j \end{pmatrix} \in \mathbb{R}^{k+l+2} : i \in \{1, \dots, k+1\}, j \in \{1, \dots, l+1\}\right\}.$$

In Figure 3 we show $\Delta_1 \times \Delta_1$ and $\Delta_2 \times \Delta_1$ for example.

Following the notation in section 2, our objects are the triangulations of $(\Delta_k \times \Delta_l, \text{vert}(\Delta_k \times \Delta_l))$, where $\text{vert}(\Delta_k \times \Delta_l)$ are the vertices. Examples of triangulations are shown in Figure 4.

First we state three lemmas later use. The volume of the $(k+l)$ -simplices in a triangulation of $\Delta_k \times \Delta_l$ is constant, which leads the following.

Lemma 4.1

The number of $(k+l)$ -simplices included in a triangulation of $\Delta_k \times \Delta_l$ is $(k+l)!/k!l!$.

The $(k+l)$ -simplices in $(\Delta_k \times \Delta_l, \text{vert}(\Delta_k \times \Delta_l))$ correspond to spanning trees in the complete bipartite graph $K_{k+1, l+1}$ [7, 7.3.D.]. This derives the next lemma.

Lemma 4.2

The number of $(k+l)$ -simplices in $(\Delta_k \times \Delta_l, \text{vert}(\Delta_k \times \Delta_l))$ is $(k+1)^l(l+1)^k$.

The problem in the lemma below can be reduced to judging the existence of a cycle in a subgraph of a directed $K_{k+1, l+1}$ (cf. [5, Lemma 2.3.]), so the time complexity follows.

Lemma 4.3

Given two maximal dimensional simplices in $\Delta_k \times \Delta_l$, judging whether their intersection is a face of each of them or not can be done in $O(k+l)$ time.

The product $\Delta_k \times \Delta_l$ has a symmetric structure: even if we commute the axes of each simplices, the shape of the product does not change. We state this symmetry formally.

Definition 4.4 (equivalence on simplices and triangulations)

Let $S_{k+1} \times S_{l+1}$ be the direct product of symmetric groups, and $(p, q) \in S_{k+1} \times S_{l+1}$.

- $S_{k+1} \times S_{l+1}$ acts on the vertices of $\Delta_k \times \Delta_l$:

$$(p, q) \begin{pmatrix} e_i \\ f_j \end{pmatrix} = \begin{pmatrix} e_{p(i)} \\ f_{q(j)} \end{pmatrix}.$$

- The action of $S_{k+1} \times S_{l+1}$ on the simplices of $(\Delta_k \times \Delta_l, \text{vert}(\Delta_k \times \Delta_l))$ is induced:

$$(p, q) \left\{ \begin{pmatrix} e_{i_1} \\ f_{j_1} \end{pmatrix}, \dots, \begin{pmatrix} e_{i_{d+1}} \\ f_{j_{d+1}} \end{pmatrix} \right\} = \left\{ \begin{pmatrix} e_{p(i_1)} \\ f_{q(j_1)} \end{pmatrix}, \dots, \begin{pmatrix} e_{p(i_{d+1})} \\ f_{q(j_{d+1})} \end{pmatrix} \right\}.$$

- The action of $S_{k+1} \times S_{l+1}$ on the triangulations of $(\Delta_k \times \Delta_l, \text{vert}(\Delta_k \times \Delta_l))$ is induced:

$$(p, q)T = \{(p, q)\sigma : \sigma \in T\}.$$

- The action of $S_{k+1} \times S_{l+1}$ on the vertices, the simplices or the triangulations defines an equivalence relation on each of them: two elements are equivalent if they can move to each other by an element of $S_{k+1} \times S_{l+1}$. We classify these sets by *orbits*: the equivalence classes.

For example, the triangulations T_1 and T_2 in Figure 4 moves to each other by $((1, 2), e) \in S_2 \times S_2$. So does T_3 and T_4 for $((1, 2), e) \in S_3 \times S_2$. The volume vectors can be regarded as a matrices

$$(\varphi_T \begin{pmatrix} e_i \\ f_j \end{pmatrix})_{ij} \in \mathbb{R}^{k+1} \times \mathbb{R}^{l+1}.$$

Those corresponding to the triangulations in Figure 4 are


$$\varphi_{T_1} = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix} \quad \varphi_{T_2} = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}.$$

$S_{k+1} \times S_{l+1}$ acts on a volume vector φ_T as rearrangements of rows and columns of a matrix. Two regular triangulations T and T' are in the same orbit if and only if their volume vectors φ_T and $\varphi_{T'}$ are in the same orbit, since the correspondence was one-to-one (cf. Theorem 2.5). We introduce an order on volume vectors, and define the representative on the orbits.


Definition 4.5 (lexicographic order on matrices)

We define lexicographic order on matrices $\mathbb{R}^{k+1} \times \mathbb{R}^{l+1}$. A matrix (a_{ij}) is smaller than (b_{ij}) if for some (i_0, j_0) , $a_{i_0 j_0} < b_{i_0 j_0}$, and for any i, j such that $i < i_0$ or such that $i = i_0$ and $j < j_0$, $a_{ij} = b_{ij}$.

Definition 4.6 (order on regular triangulations)


We introduce a total order on regular triangulations by comparing their volume vectors as matrices. 

Definition 4.7 (representative of orbits of regular triangulations)

We take the maximum triangulation as the *representative* of the orbits of regular triangulations. 

In Figure 4, T_2 becomes the representative for the orbit $\{T_1, T_2\}$.

Lemma 4.8

Given a regular triangulation T , the representative element of its orbit can be computed in $O(l! k^2 l^2)$ time. 

Proof. In order to choose a representative triangulation from the orbit of a given regular triangulation, we have to find an element of $S_{k+1} \times S_{l+1}$ whose corresponding rearrangement maximizes the matrix of the volume vector φ_T . We check all $(l+1)!$ cases of arrangement of columns. For each of them we sort the rows. There are $k+l$ rows of length $l+1$, so this takes $O((k+1)^2(l+1))$ time. Hence the whole time complexity is $O((l+1)!(k+1)^2(l+1)) = O(l! k^2 l^2)$. ■

5 Enumerating regular triangulations of $\Delta_k \times \Delta_l$

5.1 Enumerating all regular triangulations

We state the possibility of reverse search for all regular triangulations of arbitrary configurations of points, and apply this to the case of $\Delta_k \times \Delta_l$. This can be done in time linear to the number of the whole and with memory linear to the size of one.

We define two triangulations to be adjacent if they can be modified along a circuit. Take the triangulations T_3 and T_4 in Figure 4 for example. They become adjacent, because they can be modified along the circuit consisting of the vertices of the upper two tetrahedra.

Definition 5.1

The structure of reverse search for regular triangulations of $(\Delta_k \times \Delta_l, \text{vert}(\Delta_k \times \Delta_l))$ is

- $S = \{\text{regular triangulations}\}$
- $\text{Adj}(T, i) = (\text{the } i\text{-th regular triangulation which can be modified from } T \text{ along a circuit})$
- $f(T) = \begin{cases} \text{Adj}(T, i) & \text{if the largest regular triangulation } \text{Adj}(T, i) \text{ among those adjacent to } T \text{ is larger than } T \\ T & \text{otherwise} \end{cases}$

The index i in the definition of $\text{Adj}(T, i)$ is not of importance. Because regular triangulations correspond to the vertices of the secondary polytope $\Sigma(\text{vert}(\Delta_k \times \Delta_l))$, and lexicographic order is same as ordering by the inner product with a vector $(K^{k+l+2}, K^{k+l+1}, \dots, K)$ for sufficiently large K , reverse search is possible. In fact, this is a geometric version of linear programming (cf. [19, Theorem 3.7.]). This algorithm for enumerating all regular triangulations for arbitrary configurations of points is from [14]. We apply this to the case of $\Delta_k \times \Delta_l$.

Theorem 5.2 ([14])

The structure of Definition 5.1 enables reverse search. For the case of $\Delta_k \times \Delta_l$ the time complexity is $O(\binom{k+l}{k}^2 k^3 l^3 \text{LP}(kl, \binom{k+l}{k}(k+l+1)) \#\mathcal{R})$, where $\text{LP}(n, m)$ is the time required to solve a linear programming problem with m strict inequalities constraints in n variables, and \mathcal{R} is the set of regular triangulations of $\Delta_k \times \Delta_l$. The memory required is linear to the size of a triangulation.

5.2 Using symmetry

By modifying the structure of reverse search in the previous subsection, we can enumerate all orbits of regular triangulations in time proportional to their cardinality.

We define two different orbits of triangulations to be adjacent if there exist adjacent triangulations chosen from each of them. We denote the orbit, or class, which contains a triangulation T by $[T]$. Clearly, if $[T]$ and $[T']$ are adjacent, there exists a triangulation $T'' \in [T']$ which is adjacent to T . The representative of an orbit was its maximum element in the lexicographic order of volume vectors.

Definition 5.3

The structure of reverse search for the orbits of regular triangulations of $(\Delta_k \times \Delta_l, \text{vert}(\Delta_k \times \Delta_l))$ is as follows. We denote the representative triangulation of an orbit $[T]$ by T_{\max} .

- $S = \{[T] : T \text{ is a regular triangulation}\}$
- $\text{Adj}([T], i) = [\text{Adj}(T_{\max}, i)]$
- $f([T]) = [f(T_{\max})]$

Theorem 5.4

The structure of Definition 5.3 enables reverse search. The time complexity is $O\left(\binom{k+l}{k}^2 l! k^5 l^5 \text{LP}(kl, \binom{k+l}{k})(k+l+1)\right) \#(\mathcal{R}/\sim)$, where \mathcal{R}/\sim is the orbits of regular triangulations. The memory required is linear to the size of a triangulation.

Proof. For an orbit $[T]$ which does not include the root triangulation in Definition 5.1, its representative is smaller than the representative of $f([T])$. So the orbit containing the root triangulation is the unique root orbit. From any orbit, we can reach the root orbit by applying f several times. Hence, reverse search works. For time and space complexity, compare this version with the algorithm in Theorem 5.2. The extra time required for Adj and f is the time to compute the representative element, which is $O(l! k^2 l^2)$ by Lemma 4.8. The time required to compute one orbit is bounded by $O(l! k^2 l^2)$ times the time for a regular triangulation in the algorithm without considering symmetry. On the whole, the coefficient of time complexity becomes $O(l! k^2 l^2)$ times larger. In the search, we are just "jumping" to the representative, so the required memory is same. \square

6 Preparations for enumerating all triangulations

6.1 Arbitrary configuration of points

We fix some notations for use in the following sections.

Definition 6.1

Let $(\text{conv}(\mathcal{A}), \mathcal{A})$ be an arbitrary configuration of points.

- $S = \{\sigma \in 2^{\mathcal{A}} : \text{maximal dimensional simplex of } (\text{conv}(\mathcal{A}), \mathcal{A})\}$
- Two simplices in S intersect if their intersection is not a face for at least one of them.
- The intersection graph of S is a graph with S the vertices and edges between two intersecting simplices.
- $\mathcal{I} = \{I \in 2^S : \text{independent set of the intersection graph of } S\}$
- $\mathcal{M} = \{I \in 2^S : \text{maximal independent set of the intersection graph of } S\}$
- Two simplices in S are adjacent if they share a facet. For any independent set $I \in \mathcal{I}$ its adjacency graph is a graph with simplices in I the vertices and edges between two adjacent simplices.
- $\mathcal{I}_{\text{con}} = \{I \in \mathcal{I} : \text{the adjacency graph of } I \text{ is connected}\}$
- $\mathcal{M}_{\text{con}} = \{I \in \mathcal{I}_{\text{con}} : \text{maximal in set inclusion in } \mathcal{I}_{\text{con}}\}$
- $\mathcal{T} = \{\{\sigma_1, \dots, \sigma_r\} \in 2^S : \text{a set of maximal dimensional simplices in a triangulation of } (\text{conv}(\mathcal{A}), \mathcal{A})\}$

\mathcal{I}_{con} is formed by the independent sets which can be made incrementally: starting by an empty set and adding recursively a simplex which is adjacent, where we promise that any simplex can be added to the empty set. We call the elements of \mathcal{I}_{con} the *connected independent sets*.

Clearly, $\mathcal{T} \subset \mathcal{M}$ and $\mathcal{T} \subset \mathcal{M}_{\text{con}} \subset \mathcal{I}_{\text{con}}$. In subsequent sections we show algorithms to enumerate \mathcal{M} and \mathcal{I}_{con} , which leads to the enumeration of \mathcal{T} the triangulations of $(\text{conv}(\mathcal{A}), \mathcal{A})$. The difference of \mathcal{T} and \mathcal{M} or \mathcal{M}_{con} becomes a loss. This sort of thing happens for Schönhardt's polyhedron (cf. [15, 10.2.1]), but the authors do not know if this occurs for the case of $\Delta_k \times \Delta_l$.

6.2 More on $\Delta_k \times \Delta_l$

Simplices in S can be regarded as matrices in $\mathbb{R}^{k+1} \times \mathbb{R}^{l+1}$:

$$\sigma = \left\{ \begin{pmatrix} e_{i_1} \\ f_{j_1} \end{pmatrix}, \dots, \begin{pmatrix} e_{i_{k+l+1}} \\ f_{j_{k+l+1}} \end{pmatrix} \right\} \in S \longleftrightarrow \sum_{\substack{e_{i_r} \\ f_{j_r} \in \sigma}} (\delta_{ii_r} \delta_{jj_r})_{ij} \in \mathbb{R}^{k+1} \times \mathbb{R}^{l+1},$$

where δ is Kronecker's delta. Remember that we defined lexicographic order on matrices in Definition 4.5.

Definition 6.2 (equivalence on S and 2^S)

- We define the action of $S_{k+1} \times S_{l+1}$ on S by restricting to S its action on all simplices, which we defined in Definition 4.4. It defines an equivalence relation on S . We notify this relation by \sim .
- The action of $S_{k+1} \times S_{l+1}$ on 2^S is induced. Its action on $\mathcal{I}, \mathcal{M}, \mathcal{I}_{\text{con}}, \mathcal{M}_{\text{con}}$ and \mathcal{T} is defined by restriction. The actions define equivalence relations on each of the sets. We also use \sim for these relations.

Clearly, $\mathcal{T}/\sim \subset \mathcal{M}/\sim \subset \mathcal{I}/\sim \subset 2^S/\sim$ and $\mathcal{T}/\sim \subset \mathcal{M}_{\text{con}}/\sim \subset \mathcal{I}_{\text{con}}/\sim \subset \mathcal{I}/\sim \subset 2^S/\sim$.

7 Enumerating all triangulations—as a subset of maximal independent sets

7.1 Triangulations for arbitrary configurations of points

The intersection graph of maximal dimensional simplices of a point configuration was the graph with these simplices the vertices and edges between two improperly intersecting simplices. Triangulations can be regarded as a subclass of the maximal independent sets of this graph [9]. Efficient algorithms to enumerate maximal independent sets are known [11], [18]. In this section, we state the relation between triangulations and maximal independent sets, and propose a triangulation enumerating algorithm using this property. This algorithm handles arbitrary configurations of points.

First, we cite from [11] the algorithm we use for enumerating maximal independent sets. The algorithm is called the generalized Paull-Unger procedure with improvements by Tsukiyama, Ide, Ariyoshi and Shirakawa [18].

Let the set of vertices be $E = \{1, \dots, n\}$ and c the independence testing time. We define \mathcal{M}_j the family of independent sets that are maximal within $\{1, \dots, j\}$. We construct \mathcal{M}_j from \mathcal{M}_{j-1} , starting from $\mathcal{M}_0 = \{\emptyset\}$, to obtain $\mathcal{M}_n = \mathcal{M}$. For each I in \mathcal{M}_{j-1} , we test the independency of $I \cup \{j\}$. If it is independent, we add it to \mathcal{M}_j . If not independent, we add I and other maximal independent sets of \mathcal{M}_j included in $I \cup \{j\}$. If I' is such set, it should be maximal in $I \cup \{j\}$. We use this fact reversely: first list up the maximal independent sets in $I \cup \{j\}$, and check if they are in \mathcal{M}_j . The algorithm elaborates to produce I' from a single I . We show it in Figure 5.

This computation performs a search on a tree. Nodes at level j correspond to members of \mathcal{M}_j with the tree rooted by \emptyset . For each I in \mathcal{M}_{j-1} , the corresponding I' (possibly several) in \mathcal{M}_j are its children. We start with the root \emptyset . Several searching methods are possible, but we take depth first search here.

Theorem 7.1 ([11])

The algorithm in Figure 5 enumerates all maximal independent sets in $O(nc'K + n^2cKK')$ time and $O(nK')$ memory. Here $K = \#\mathcal{M}$ and we suppose that in *Step 1*, for each $I \in \mathcal{M}_{j-1}$, at most K' sets I' are found in c' time.

Now we apply the algorithm above to our case.

Theorem 7.2

If we have $E = S$ with an arbitrary fixed order with an oracle that answers the previous or next simplex for a given one in unit time, and apply the algorithm in Figure 5 to enumerate all maximal independent sets of the intersection graph of S , it works in $O(m \text{time}(\text{intersect})(\#S)^2 \#\mathcal{M})$ time with the memory for the size of one triangulation. Here $m = \max_{I \in \mathcal{M}} \#I$ is the maximum cardinality of maximal dimensional simplices in \mathcal{M} and $\text{time}(\text{intersect})$ is the time to judge if two simplices intersect properly.

Proof. For the independence test, or the test in *Step 1*, in actual we only have to check the intersection of a newly added simplex with the less than m current ones in I , so c and c' in Theorem 7.1 is computed in $m \cdot \text{time}(\text{intersect})$ time. In *Step 1*, for each I in \mathcal{M}_{j-1} , if $I \cup \{j\} \in \mathcal{M}_j$, $I' = I \cup \{j\}$, and if not we take I and the set of simplices in $I \cup \{j\}$ except those intersecting with j as applicants, so $\#K' \leq 2$. Since we have the oracle mentioned above, we can traverse the search tree only with the information of our current independent set I and depth j , which is practically same as the size of a triangulation. Furthermore, since backtracking is easy, the order of time complexity does not change. \square

7.2 Triangulations for $\Delta_k \times \Delta_l$

We apply Theorem 7.2 to the case of $\Delta_k \times \Delta_l$.

Theorem 7.3

For the point configuration $(\Delta_k \times \Delta_l, \text{vert}(\Delta_k \times \Delta_l))$ the algorithm in Figure 5 enumerates all maximal independent sets of the intersection graph of S in $O(\binom{k+l}{k}(k+l)k^{2l}l^{2k}\#\mathcal{M})$ time with the memory for the size of a triangulation.

Proof. The simplices in $\Delta_k \times \Delta_l$ correspond to spanning trees of the bipartite graph $K_{k+1, l+1}$. We can generate such trees in unit time ([10], [16]), so the oracle mentioned in the Theorem 7.2 exists. Because $\#S = (k+1)^l(l+1)^k$ from Lemma 4.2, and checking the linearity of vertices from $\text{vert}(\Delta_k \times \Delta_l)$ can be done in $O(k+l)$ time analogously to Lemma 4.3, we can list all maximal dimensional simplices S in $O((k+l)(k+1)^{l+1}(l+1)^{k+1})$ time. By Lemma 4.1, $m = \binom{k+l}{k}$, and by Lemma 4.3 $\text{time}(\text{intersect}) = O(k+l)$. \square

8 Enumerating all triangulations—as a subset of connected independent sets

8.1 Triangulations for arbitrary configurations of points

We propose an algorithm to enumerate all independent sets with a connected adjacency graph, i.e. with a connected internal, for arbitrary configurations of points. All triangulations are included in this class. Reverse search is used, so the time complexity is proportional to the number of those independent sets, and memory for only twice the size of a triangulation is required.

Our aim here is to enumerate \mathcal{I}_{con} , which leads to the enumeration of the triangulations \mathcal{T} of $(\text{conv}(\mathcal{A}), \mathcal{A})$. Imagine the Hasse diagram of 2^S . The induced subgraph for the vertices $\mathcal{I}_{\text{con}} \subset 2^S$ is connected. We construct a spanning tree in this induced subgraph and enumerate the vertices of \mathcal{I}_{con} . We assume a total order on S , and the induced lexicographic order on 2^S . Any order on S is possible. For each $I \in \mathcal{I}$ the incrementable facets are the facets of a simplex in I included in $\partial(\bigcup_{\sigma \in I} \text{conv}(\sigma)) \setminus \partial(\text{conv}(\mathcal{A}))$, where ∂P is the union of facets of P . Let m_{facet} be the maximum cardinality of incrementable facets for all $I \in \mathcal{I}_{\text{con}}$. Fix a total order on the incrementable facets of each $I \in \mathcal{I}_{\text{con}}$, and on vertices \mathcal{A} .

Theorem 8.1

For the following structure, reverse search enumerates all sets in \mathcal{I}_{con} , in which all triangulations are included.

- $\delta = m_{\text{facet}} \# \mathcal{A} + 1$
- $f(I) = \max_{\text{order on } 2^S} \{I \setminus \{\sigma\} : \sigma \in I\}$
- $\text{Adj}(I, (i+1)\# \mathcal{A} + j) = \begin{cases} I \cup \{\text{conv}(\tau_i \cup \{v_j\})\} & \text{for the } i\text{-th incrementable facet } \tau_i \text{ of } I \text{ and } j\text{-th point } v_j \in \mathcal{A}, \text{ if } \tau_i \cup \{v_j\} \text{ is a maximal dimensional simplex, and does not intersect with or is not equal to the simplices in } I. \\ f(I) & \text{if } (i+1)\# \mathcal{A} + j = \delta \\ \emptyset & \text{otherwise} \end{cases}$
- $\text{Adj}(\emptyset, i) = \{\text{the } i\text{-th simplex in } \mathcal{S}\}$
- $\tau = \emptyset \in \mathcal{I}_{\text{con}}$

The time complexity is $O((m_{\text{facet}} \# \mathcal{A} + 1)m \text{time}(\text{intersect}) \# \mathcal{I}_{\text{con}})$, where $m = \max_{I \in \mathcal{M}} \# I$ is the maximum cardinality of maximal dimensional simplices in \mathcal{M} and $\text{time}(\text{intersect})$ is the time to judge if two simplices intersect properly. The memory required is the size of two triangulations.

Proof. Slight modifications are required for the case of the empty set. We define all simplices in \mathcal{S} to be adjacent to \emptyset . The degree of \emptyset can be larger than δ , but we can check that this does not change the whole time complexity. Clearly, reverse search for this structure works. We can keep the elements of I sorted, so $\text{time}(f)$ is constant. To compute an adjacent vertex, we have to check if $\tau_i \cup \{v_j\}$ is a simplex, and if so, if it intersects properly with the existing simplices in I . The second test takes $m \text{time}(\text{intersect})$, and generally the first one can be done in $\text{time}(\text{intersect})$ time. Thus $\text{time}(\text{Adj})$ is $O(m \text{time}(\text{intersect}))$. \square

We do not know a non-trivial bound for the gap between $\# \mathcal{M}_{\text{con}}$ and $\# \mathcal{I}_{\text{con}}$. This gap becomes a loss for the time complexity, compared to the cardinality of triangulations, so the evaluation of this gap is important to decide the efficiency of this algorithm.

8.2 Triangulations for $\Delta_k \times \Delta_l$

We apply the algorithm above to the case of $\Delta_k \times \Delta_l$.

Theorem 8.2

The algorithm in Theorem 8.1 enumerates all connected independent sets, in which all triangulations are included, in $O\left(\binom{k+l}{k}^2 (k+l)^2 kl \# \mathcal{I}_{\text{con}}\right)$ time with the memory for the size of two triangulations.

Proof. Recall Lemma 4.1. Less than $\binom{k+l}{k}$ simplices appear in an independent set, and each $(k+l)$ -simplex has $k+l+1$ facets, so $m_{\text{facet}} \leq \binom{k+l}{k} (k+l+1)$. The cardinality of points \mathcal{A} is $(k+l)(l+1)$. Judging whether two simplices intersect or not can be done in $O(k+l)$ time (Lemma 4.3). \square

The figure $\binom{k+l}{k}$ here seems to be large, but this was the cardinality of maximal simplices appearing in a triangulation.

why?

to sketchy

8.3 Using symmetry

We can modify the algorithm in Theorem 8.1 to enumerate all classes of triangulations of $(\Delta_k \times \Delta_l, \text{vert}(\Delta_k \times \Delta_l))$ with respect to the symmetry of $S_{k+1} \times S_{l+1}$.

As stated above Theorem 8.1, we can introduce any order on \mathcal{S} , and this is true also for the rest of this paper. But, from now on, we take the lexicographic order of its corresponding matrix representation (cf. Definition 4.5, subsection 6.2). It is easy to reformulate the following to an arbitrary order.

We define a label for each element in \mathcal{I}_{con} . By the lexicographic order on the labels, we introduce a total order on \mathcal{I}_{con} . The maximal element in this order will be taken as the representative for each class in $\mathcal{I}_{\text{con}}/\sim$.

Definition 8.3 (labeling on \mathcal{I}_{con})

The labeling $l : \mathcal{I}_{\text{con}} \rightarrow (\mathbb{R}^{k+1} \times \mathbb{R}^{l+1})^*$, where Σ^* is the set of words on an alphabet Σ , is an alignment of simplices in a connected independent set $I \in \mathcal{I}_{\text{con}}$. We align the simplices by the order we visit in performing a breadth first search on the adjacency graph of I . We start from the maximum simplex in I , and when several simplices are adjacent to our current simplex, we visit from the larger ones.

For the connected independent set I in Figure 6 which is an example from $\Delta_2 \times \Delta_2$, the labeling is

$$l(I) = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}.$$

Clearly, l is injective.

From the order on the matrices $\mathbb{R}^{k+1} \times \mathbb{R}^{l+1}$, a lexicographic order on $(\mathbb{R}^{k+1} \times \mathbb{R}^{l+1})^*$ is induced. The order between words having different length does not matter in our case.

Definition 8.4 (order on \mathcal{I}_{con})

We define a total order on \mathcal{I}_{con} by the order of their labels in $(\mathbb{R}^{k+1} \times \mathbb{R}^{l+1})^*$.

Definition 8.5 (representative of $\mathcal{I}_{\text{con}}/\sim$)

For each class in $\mathcal{I}_{\text{con}}/\sim$, we take the maximum element in this order as the representative.

Now we define the parent function f for reverse search.

Definition 8.6

The parent function f on the connected independent sets \mathcal{I}_{con} is, for $I \in \mathcal{I}_{\text{con}}$ with label $l(I) = \sigma_1 \cdots \sigma_k$, $f(I) = \{\sigma_1, \dots, \sigma_{k-1}\}$. We promise $f(\emptyset) = \emptyset$.

By executing breadth first search in parallel, we can prove the following lemma.

Lemma 8.7

If a connected independent set $I \in \mathcal{I}_{\text{con}}$ is a representative for its class $[I] \in \mathcal{I}/\sim$, $f(I)$ is the representative for its class $[f(I)] \in \mathcal{I}_{\text{con}}/\sim$.

Using this property, we can enumerate all classes in $\mathcal{M}_{\text{con}}/\sim$ by tracking their representatives.

Here is the version for classes of connected independent sets.

Theorem 8.8

If we take δ and r same as the case of $\Delta_k \times \Delta_l$ for Theorem 8.1, f as in Definition 8.6 and Adj to return the adjacent vertex only when it is a representative, reverse search enumerates all representative elements of $\mathcal{I}_{\text{con}}/\sim$, which includes all representatives for triangulations \mathcal{T}/\sim , of $\Delta_k \times \Delta_l$. The time complexity is $O(\binom{k+l}{k}^2 (k+l)k^3 l^3 k! l! \#(\mathcal{I}_{\text{con}}/\sim))$, and required memory is for the size of two triangulations.

Proof. Because of Lemma 8.7, reverse search works. We only have to check the time complexity of Adj. For each adjacent vertex in Theorem 8.1, we have to check if it is a representative. There are $(k+1)!(l+1)!$ cases corresponding to $S_{k+1} \times S_{l+1}$, and for each case, we have to sort the vertices and execute breadth first search to calculate its label. This takes $O(\binom{k+l}{l}^2 (k+1)(l+1))$ time, where $\binom{k+l}{k}$ was the maximum cardinality of maximal dimensional simplices i.e. the maximum number of vertices in the adjacency graph. Thus time(Adj) changes from $O(m \text{ time}(\text{intersect})) = O(\binom{k+l}{k} (k+l))$ to $O(\binom{k+l}{k}^2 (k+1)!(l+1)!(k+1)(l+1))$. \square

References

- [1] DAVID AVIS & KOMEI FUKUDA: *A Pivoting Algorithm for Convex Hulls and Vertex Enumeration of Arrangements and Polyhedra*, *Discrete Comput. Geom.* **8** (1992), 295–313
- [2] DAVID AVIS & KOMEI FUKUDA: *Reverse Search for Enumeration*, *Discrete Appl. Math.* **65** (1996), 21–46
- [3] LOUIS J. BILLERA, PAUL FILLIMAN & BERND STURMFELS: *Constructions and Complexity of Secondary Polytopes*, *Advances in Math.* **83** (1990), 155–179
- [4] JESÚS A. DE LOERA: *Computing Regular Triangulations of Point Configurations*, 1994
ftp://cam.cornell.edu/pub/puntos/help.ps
- [5] JESÚS A. DE LOERA: *Nonregular Triangulations of Products of Simplices*, *Discrete Comput. Geom.* **15** (1996), 253–264
- [6] JESÚS A. DE LOERA, SERKAN HOŞTEN, FRANCISCO SANTOS & BERND STURMFELS: *The Polytope of All Triangulations of a Point Configuration*, *Doc. Math.* **1** (1996), 103–119
- [7] ISRAEL M. GELFAND, MIKHAIL M. KAPRANOV & ANDREI V. ZELEVINSKY: *Discriminants, Resultants and Multidimensional Determinants*, Birkhäuser, Boston 1994
- [8] ISRAEL M. GEL'FAND, ANDREI V. ZELEVINSKIĬ & MIKHAIL M. KAPRANOV: *Newton Polyhedra of Principal A-determinants*, *Soviet Math. Dokl.* **40** (1990), 278–281
- [9] HIROSHI IMAI & KEIKO IMAI: *Triangulation and Convex Polytopes*, in: “Geometry of Toric Varieties and Convex Polytopes”, *RIMS Kokyuroku* **934** (1996), Research Institute for Mathematical Sciences, Kyoto University, 149–166 (in Japanese)
- [10] H. N. KAPOOR & H. RAMESH: *Algorithms for Generating All Spanning Trees of Undirected, Directed and Weighted Graphs*, *Lecture notes in Computer Science*, Springer-Verlag, 1992, 461–472
- [11] E. L. LAWLER, J. K. LENSTRA & A. H. G. RINNOOY KAN: *Generating All Maximal Independent Sets: NP-Hardness and Polynomial-Time Algorithms*, *SIAM J. Comput.* **9** (1980), 558–565
- [12] CARL W. LEE: *Regular Triangulations of Convex Polytopes*, in: “Applied Geometry and Discrete Mathematics—The Victor Klee Festschrift” (Peter Gritzmann and Bernd Sturmfels, eds.), *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **4**, Amer. Math. Soc. 1991, 443–456
- [13] TOMONARI MASADA: *An Algorithm for the Enumeration of Regular Triangulations*, Master's Thesis, Department of Information Science, University of Tokyo, March 1995
- [14] TOMONARI MASADA, HIROSHI IMAI & KEIKO IMAI: *Enumeration of Regular Triangulations*, in: “Proceedings of the Twelfth Annual Symposium on Computational Geometry” Association for Computing Machinery (ACM), New York 1996, 224–233
- [15] JOSEPH O'ROURKE: *Art Gallery Theorems and Algorithms*, *International Series of Monographs on Computer Science* **3**, Oxford University Press, New York 1987
- [16] A. SHIOURA, A. TAMURA & T. UNO: *An Optimal Algorithm for Scanning All Spanning Trees of Undirected Graphs* *SIAM J. Comp.*, to appear
- [17] BERND STURMFELS: *Gröbner Bases of Toric Varieties*, *Tôhoku Math. J.* **43** (1991), 249–261
- [18] SHUJI TSUKIYAMA, MIKIO IDE, HIROMU ARIYOSHI & ISAO SHIRAKAWA: *A New Algorithm for Generating All the Maximal Independent Sets* *SIAM J. Comput.* **6** (1977), 505–517
- [19] GÜNTER M. ZIEGLER: *Lectures on Polytopes*, *Graduate Texts in Mathematics* **152**, Springer-Verlag, New York 1995

Appendices

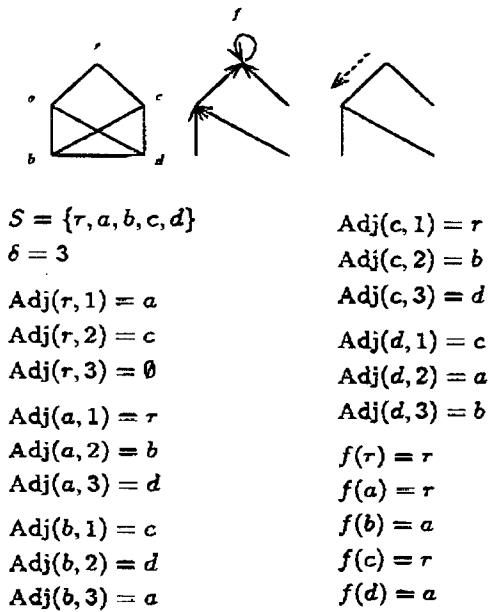


Figure 1: An example of a set and its adjacency (top left), parent-children relation (top center), the reverse search tree (top right) and the structure in formulas (above)

```

ReverseSearch( $\delta, Adj, f, r$ )
 $v := r \quad j := 0$ 
repeat
  while  $j < \delta$  do
     $j := j + 1 \quad next = Adj(v, j)$ 
    if  $next \neq \emptyset$  then
      if  $f(next) = v$  then
         $\{v := next \quad j := 0\}$ 
  if  $v \neq r$  then
     $u := r \quad v := f(v)$ 
     $j := 0$ 
    repeat  $j := j + 1$ 
    until  $Adj(v, j) = u$ 
until  $v = r$  and  $j = \delta$ 
    
```

Figure 2: The algorithm of reverse search

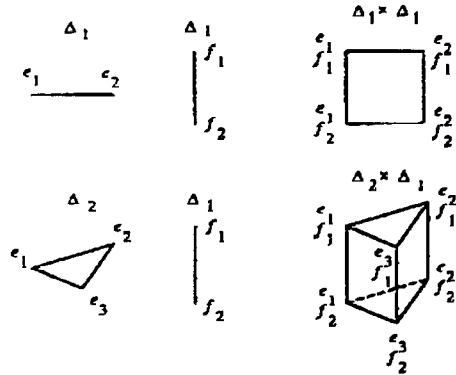


Figure 3: Product of simplices: $\Delta_1 \times \Delta_1$ and $\Delta_2 \times \Delta_1$

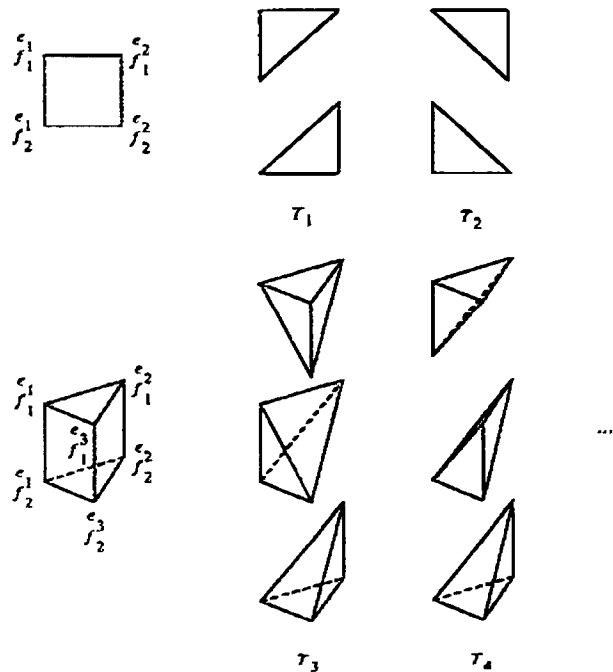


Figure 4: Triangulations for $\Delta_1 \times \Delta_1$ and $\Delta_2 \times \Delta_1$

Step 1. For each $I \in \mathcal{M}_{j-1}$, find all independent sets I' that are maximal within $I \cup \{j\}$.

Step 2. For each such I' , test I' for maximality within $\{1, \dots, j\}$. Each set I' that is maximal within $\{1, \dots, j\}$ is a member of \mathcal{M}_j , and each member of \mathcal{M}_j can be found in this way. However a given $I' \in \mathcal{M}_j$ may be obtained from more than one $I \in \mathcal{M}_{j-1}$. In order to eliminate duplications we need one further step.

Step 3. For each I' obtained from $I \in \mathcal{M}_{j-1}$ that is maximal within $\{1, \dots, j\}$, test for each $i < j$, $i \notin I'$, the set $(I' \setminus \{j\}) \cup (I' \cap \{1, \dots, i-1\}) \cup \{i\}$ for independence. Reject I' if any of these tests yields an affirmative answer. (This step retains I' only if it is obtained from the lexicographically smallest $I \in \mathcal{M}_{j-1}$.)

Figure 5: The algorithm for enumerating maximal independent sets

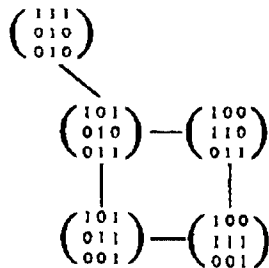


Figure 6: An adjacency graph of a connected independent set from the case for $\Delta_2 \times \Delta_2$