Enumerating geometric and combinatorial objects by reverse search

Shin-ichi Tanigawa
RIMS, Kyoto University

November 10, 2011

1 Reverse search

1.1 How to design an enumeration algorithm

Let V be a finite set V with $n = |V|$. We shall consider the problem of enumerating a family $\mathcal{O}$ of subsets of V which have a specified combinatorial structure. For example, $V = \{1, 2, \ldots, n\}$ and $\mathcal{O}$ is the set of all permutations, or V is an edge set of a graph and $\mathcal{O}$ is a family of (edge sets of) spanning trees in the graph. Below we list the procedure for enumerating $\mathcal{O}$ by reverse search.

Step 1: Define a connected search graph on $\mathcal{O}$.

Step 1.1: Define a *flip* operation,

- that is, a (simple) operation which generates a new object $o_1 \in \mathcal{O}$ from the current object $o_2 \in \mathcal{O}$. E.g., swap operations on permutations; diagonal flips on triangulations.
- A flip operation is called a *k-flip* if the symmetric difference between o_1 and o_2 is at most $2k$.

Step 1.2: Define the search graph $\mathcal{G} = (\mathcal{O}, E)$

- where vertices are corresponding to the objects and o_1 and o_2 are adjacent iff they can be converted to each other by a single flip.

Step 1.3: Prove $\mathcal{G}$ is connected.

Step 2: Define a search tree in $\mathcal{G}$.

Step 2.1: Define a unique *root* of $\mathcal{G}$ and the *parent* for each non-root node of $\mathcal{G}$.

- The parent should be defined by using only the local information.
- The function which returns a parent is called a *parent function.*

Step 2.2 Prove the parent-child relation forms a rooted tree (search tree) in $\mathcal{G}$.

- Typical example: Define a potential $p : \mathcal{O} \to \mathbb{R}$ which is injective and can be computed without looking at the other nodes. Then, define a root r to be $\mathrm{argmax}_{o \in \mathcal{O}} p(o)$ and the parent $\mathrm{argmax}_{o' \in N(o)} p(o')$ for each $o \in \mathcal{O} \setminus \{r\}$.
- A potential may be defined from $\mathcal{O}$ to a totally ordered set, e.g, the set of k-tuples of indices with lexicographical ordering.

Step 3: Trace the search tree from the root by applying the reverse search technique.

- Compute the root r and set $v := r$.
- Repeat the following until $v = r$ and all neighbors are "checked".
  - ($\star$) For each $u \in N(v)$,
    - ask the parent of u;
    - if the parent of u is v;
    - then output u and go forward to u (i.e., recursively go to ($\star$) by setting $v := u$)
  - (After testing all neighbors) backtrack to the parent.

The total computational time is roughly $n^{O(k)}|\mathcal{O}|$ since $|N(v)| = O(n^{2k})$. So, what we have to do is

- to define a k-flip operation with small k so that the search graph becomes connected;
- to define an easily computable parent function so that the parent-child relation forms a rooted tree;
- an efficient implementation.

1.2 Example we have seen in the last lecture: Enumeration of triangulations

Let us look at the problem for enumerating all triangulations on a given point set in the plane (in general position). Recall the following terms and fundamental theorems:

- Delaunay triangulation (shortly DT): a triangulation which satisfies the empty circle property (i.e., for each edge $ab \in DT$ there is an empty circle passing through a and b).
- Diagonal flip: an operation that replace an edge with another diagonal edge within a quadrilateral in T.
- Illegal edge: an edge ab which is incident to two triangles, say abc and acd in T and the circumcircle of abc does not contain d in its interior.
- Angle vectors: the vector of interior angles sorted in non-decreasing order.

Proposition 1 (See e.g. [4]). *Suppose P is in general position.*

(i) *There exists a unique Delaunay triangulation DT on P.*

(ii) *DT has the lexicographically maximum angle vector among all triangulations on P.*

(iii) *If a triangulation is not Delaunay, then there always exists an illegal edge.*

(iv) *Moreover, the diagonal flip on an illegal edge decreases the angle vector.*

(v) *T can be converted to DT by greedily flipping illegal edges.* □

This gives us all ingredients:

- Flip: diagonal flip (1-flip);
- Root: the Delaunay triangulation;
- Potential: angle vectors:

- Parent: the triangulation which has the lexicographically maximum angle vector among the neighbors (that is, triangulations obtained by a diagonal flip).

Then the remaining task is to prove (i) the search graph defined by diagonal flips is connected and (ii) the parent-child relation indeed forms a rooted tree. But we are already done: for a non-Delaunay triangulation T Proposition 1 implies that there always exists a triangulation which can be moved from T by a diagonal flip and whose angle vector is lexicographically smaller than that of T; Thus, the parent of T always has the smaller angle vector and tracing the parents we finally reach to a Delaunay triangulation, which is our root; This implies that the parent-child relation forms a rooted-tree and simultaneously implies that the search graph is connected.

2 Enumeration of bases of matroids

The next problem is the enumeration of spanning trees of graphs. But here we shall present a much abstract approach by generalizing the concept of spanning trees. We are going to develop an enumeration algorithm for bases of *matroids*. Let's start with the definition of matroids. (For details, see a book [5] of matroid theory.)

2.1 Matroids

A matroid is an abstraction of a matrix, which unifies two concepts of linear algebra and graph theory, the linear independence in vector spaces and cycle-freeness in undirected graphs. Let us consider the set $V = \{v_1, v_2, \ldots, v_n\}$ of vectors in a finite dimensional vector space. Let $\mathcal{I}$ be the family of linearly independent subsets of V. Then, observe that $\mathcal{I}$ satisfies the following properties:

(I2) $\forall Y \in \mathcal{I}$ and $\forall X \subseteq Y, X \in \mathcal{I}$;

(I3) $\forall X, Y \in \mathcal{I}$ with $|X| < |Y|$, $\exists x \in Y$ s.t. $X + x \in \mathcal{I}$.

Recall basic facts from linear algebra: Any subset of a linearly independent set is also linearly independent, which is (I2). (I3) can be verified as follows: $\dim(\mathrm{span}(X)) = |X| < |Y| = \dim(\mathrm{span}(Y)) \Rightarrow \exists x \in Y$ with $x \notin \mathrm{span}(X) \Rightarrow X + x$ is independent. Also we may say that the empty set is always independent:

(I1) $\emptyset \in \mathcal{I}$.

A matroid is defined by extracting these three properties (I1)(I2)(I3) as axioms.

Definition 1. *A* matroid *is a pair $\mathcal{M} = (E, \mathcal{I})$ of a finite set E and a family $\mathcal{I}$ of subsets of E satisfying (I1)(I2)(I3). E is called the* ground set *and an element in $\mathcal{I}$ is called an independent set.*

We list examples:

- *linear matroid*
  - E: the set of vectors.
  - $\mathcal{I}$: the family of linearly independent subsets of E.
- *uniform matroid $U_{n,m}$*
  - E: a finite set of n elements.
  - $\mathcal{I} := \{I \subseteq E \mid |I| \leq m\}$.

- *graphic matroid* (cycle matroid)
  - E: the edge set of a graph G.
  - $\mathcal{I}$: the family of forests (i.e., cycle-free edge subsets) in G.
- The following $(E, \mathcal{I})$ may not be a matroid.
  - E: the edge set of a graph G.
  - $\mathcal{I}$: the family of matchings in G.
- *matching matroid* $(V, \mathcal{I})$
  - V: the set of vertices of a graph G.
  - $\mathcal{I} := \{I \subseteq V \mid I \text{ is the set of end-vertices of some matching in } G\}$.

2.2 Basics of matroids

Definition 2. *A maximal independent set of a matroid $\mathcal{M}$ is called a* base *of $\mathcal{M}$.*

For example, if $\mathcal{M}$ is a linear matroid, a base is a basis of the underlying vector space; if $\mathcal{M}$ is a graphic matroid, a base is a spanning tree of an associated connected graph.

Let us denote by $\mathcal{B}$ the family of all bases of $\mathcal{M}$. $\mathcal{B}$ has the following nice properties.

Lemma 2. *All bases of a matroid $\mathcal{M} = (E, \mathcal{I})$ have the same cardinality.*

Proof. Suppose $|B_1| < |B_2|$ for $B_1, B_2 \in \mathcal{B}$. Then there exists $x \in B_2 \setminus B_1$ with $B_1 + x \in \mathcal{I}$ by (I3). This contradicts the maximality of B_1. $\square$

Theorem 3. *$\mathcal{B}$ satisfies the following properties:*

(B1) $\mathcal{B} \neq \emptyset$.

(B2) $\forall B_1, B_2 \in \mathcal{B}$ *and* $\forall x \in B_1 \setminus B_2$, $\exists y \in B_2 \setminus B_1$ *s.t.* $B_1 - x + y \in \mathcal{B}$.

Proof. (B1) follows from (I1). To see (B2), observe $|B_1 - x| < |B_2|$ by Lemma 2. Hence there exists $y \in B_2 \setminus (B_1 - x)$ such that $B_1 - x + y \in \mathcal{I}$ by (I3). Note $x \neq y$. By Lemma 2 again, we conclude $B_1 - x + y \in \mathcal{B}$. $\square$

The operation of (B2) is often referred to as a *base exchange operation.* Conversely, conditions (B1)(B2) are enough for characterizing matroids:

Theorem 4. *For a finite set E, the family $\mathcal{B}$ of subsets of E forms the base family of a matroid if and only if $\mathcal{B}$ satisfies (B1) and (B2).*

The proof can be found in [5]. So (B1)(B2) give another axioms which defines matroids. Namely, we may define a matroid as a pair $(E, \mathcal{B})$ of a finite set E and a family of subsets satisfying (B1)(B2) instead of (I1)(I2)(I3). Here is another interesting matroid defined in terms of $(E, \mathcal{B})$:

- *rigidity matroid*
  - E: the set of bars of a rigid pin-jointed truss structure (in structural engineering)
  - $\mathcal{I}$: the family of bar-subsets each of which forms a minimally rigid sub-structure connecting all joints.

Matroids have a quite nice property even from an optimization point of view, which we should remark here:

Theorem 5. *Let $\mathcal{M} = (E, \mathcal{I})$ be a matroid on a finite set $E = \{e_1, \ldots, e_n\}$ and $w : E \to \mathbb{R}$ be a weight function on E. Then, the maximum weight independent set can be computed by the following greedy algorithm:*

(i) *Set $X_0 = \emptyset$ and relabel the elements $\{e_1, \ldots, e_n\}$ in the non-increasing order of weight.*

(ii) *For $i = 1, 2, \ldots, n$, do the following: if $X_{i-1} + e_i \in \mathcal{I}$ and $w(e_i) > 0$ then $X_i := X_{i-1} + e_i$; otherwise $X_i := X_{i-1}$.*

(iii) *Output X_n.*

Proving Theorem 5 is a good exercise. Rather surprisingly, matroids almost characterize discrete optimization problems which can be solved by greedy algorithm (see [5] for the proof).

Theorem 6. *A pair $\mathcal{M} = (E, \mathcal{I})$ is a matroid if and only if (I1)(I2) hold and the greedy algorithm works for fining a maximum weight set in $\mathcal{I}$ for any weight function.*

2.3 Enumeration of bases of a matroid

Let us go back to the enumeration problem. Out problem is to find all bases of a matroid on a given ground set E. Notice that this problem includes

- the problem of enumerating all spanning trees in a given connected graph $G = (V, E)$;
- the problem of enumerating all column bases of the colum space of a matrix;
- the problem of enumerating all minimally rigid truss structures connecting a given joint set.

As before, we have to define a root, a flip, and a parent function. To do that, denote the ground set by $E = \{e_1, e_2, \ldots, e_n\}$ and take a base B^*. Let $r = |B^*|$, and without loss of generality we assume $B^* = \{e_1, e_2, \ldots, e_r\}$ (otherwise we can relabel the elements). The label vector $\ell(B)$ of a base B is defined as the vector of r labels of all elements in B sorted in increasing order. Then we define ingredients as follows:

- Root: B^*;
- Flip: base exchange operation (1-flip);
- Parent: a base which can be obtained by a flip and has more elements of B^* than the current. (If there are many such bases, we select one of them following some fixed rule, say, the lexicographical ordering of the label vectors $\ell(B)$.)

Then the remaining task is to prove the following:

Claim 7. *The above parent function is well-defined and the parent-child relation indeed forms a rooted-tree.*

Proof. Consider $B \in \mathcal{B}^*$ with $B \neq B^*$. Take any edge $e \in B \setminus B^*$. By (B3), there exists $f \in B^* \setminus B$ with $B - e + f \in \mathcal{B}$. Since $|(B - e + f) \cap B^*| > |B \cap B^*|$, there always exists a base in the neighbors of B that contains more elements of B^* than B. Thus, the parent function is well-defined.

Tracing parents from any base, we clearly reach to B^*. This implies that the parent-child relation forms a rooted-tree. □

3 Enumeration of non-crossing spanning trees

So far, our reverse search algorithms are based on the well-developed theorems from geometry or combinatorics. However, one of interesting job in the design of enumeration algorithms is to prove the connectivity of search graph by ourselves. Here we shall give an example for non-crossing spanning trees.

Given a point set P in the plane, a spanning tree on P is called *non-crossing* if any two edges do not intersect at a point except their endpoints. As we have seen above, the set of spanning trees is connected by 1-flips (base-exchange operations). The set of non-crossing spanning trees is however not the base family of a matroid in general, and it may not be true that the set of non-crossing spanning trees is not connected by 1-flips. Here we shall prove the connectivity based on the approach developed in [2].

A triangulation which contains a specified non-crossing edge set F is called an *F-constrained triangulations*. Recall that all results on the Delaunay triangulation can be extended to the case of F-constrained triangulations (see e.g. [4]):

- An F-illegal edge is an illegal edge not contained in F.
- The F-constrained Delaunay triangulation (shortly $DT(F)$) is the triangulation which has the lexicographically maximum angle vector among all F-constrained triangulations on P.
- If an F-constrained triangulation is not F-constrained Delaunay, then there always exists an F-illegal edge. Thus F-constrained T can be converted to $DT(F)$ by greedily flipping F-illegal edges. □

Let $\mathcal{S}$ be the set of all non-crossing spanning trees on P and let $\mathcal{S}^* = \{S \in \mathcal{S} \mid DT(S) = DT\}$. Our method is based on a trivial observation: Any spanning tree in a triangulation is non-crossing since a triangulation is a plane graph. We already learned how to enumerate all elements of $\mathcal{S}^*$ efficiently — this is just the enumeration of all spanning trees in a triangulation. Our idea is to measure how far $S \in \mathcal{S}$ is from $\mathcal{S}^*$. To do this we define the angle vector $v(S)$ of a non-crossing spanning tree S to be that of the S-constrained Delaunay triangulation $DT(S)$.

Theorem 8. *The set of non-crossing spanning trees on a point set in the plane is connected by 1-flip operations.*

Proof. Let $\mathcal{M}$ be a graphic matroid on the edge set of DT. Then, as noted above, the set of bases of $\mathcal{M}$ is equal to $\mathcal{S}^*$. By Claim 7, the set of bases is connected by base-exchange operations (that is, 1-flips); in particular, any spanning tree can be converted to the minimum weight spanning tree by a sequence of 1-flips.

Thus let us consider a non-crossing spanning tree S with $DT(S) \neq DT$. Since $DT(S) \neq DT$ but every edge of $DT(S)$ cannot be illegal except for edges in S, there is an illegal edge $e \in S$ in $DT(S)$. So, after relaxing the constraint on e, the diagonal flip on e will occur. Namely, the angle vector of $DT(S-e)$ will be strictly larger than that of $DT(S)$. Since $DT(S-e)$ is a connected graph, there exists an edge $f \in DT(S-e)$ which connects two connected components of $S-e$. Moreover $S-e+f$ is non-crossing spanning tree since it is contained in $DT(S-e)$. Since $DT(S-e) = DT(S-e+f)$ (why?) and $DT(S-e)$ has a larger angle vector than $DT(S)$, we found that $v(S-e+f)$ is the lexicographically smaller than $v(S)$.

If we repeat the above process, $v(S)$ will be minimized; in other words S is converted to an element of $\mathcal{S}^*$ by a sequence of 1-flips. □

Let S^* be the Euclidean minimum spanning tree. Recall that the Euclidean minimum spanning tree is contained in the Delaunay triangulation. If we define a potential $p(S)$ of a non-crossing spanning tree S by a tuple $(v(S), |S \cap S^*|, \ell(S))$ of the angle vector $v(S)$, $|S \cap S^*|$, and the label vector $\ell(S)$, then S^* has the lexicographically maximum potential. Moreover, the above proof implies that, if S is not the Euclidean minimum spanning tree, then there always exists a non-crossing spanning tree S' which can be obtained from S by a 1-flip and has a larger potential. Consequently, if we define the parent of S by S', the parent-child relation forms a rooted tree, and thus we can enumerate all non-crossing spanning trees by reverse search.

The same approach can be applied to the enumeration of all non-crossing minimally rigid truss structures connecting a given joint set because the method relies only on the base-exchange property of spanning trees (see [2]). Also, if you get interested in flip operations or enumerations of non-crossing geometric graphs, please check [6] and [3].

Open Problems

- Enumerate all non-crossing perfect matching on a point set P in the plane with $|P| = 2n$ in polynomial time per output.
- Enumerate all simple polygons on a point set P in the plane in polynomial time per output.

4 Exercise

The exercise is to develop a faster algorithm for enumerating bases of a matroid $\mathcal{M} = (E, \mathcal{B})$. It is known that the family of bases satisfies another-type of base exchange property: (Compare with (B2))

(B2') $\forall B_1, B_2 \in \mathcal{B}$ and $\forall y \in B_2 \setminus B_1$, $\exists x \in B_1 \setminus B_2$ s.t. $B_1 - x + y \in \mathcal{B}$.

In this exercise you may use this fact without proof.

Consider a matroid $\mathcal{M} = (E, \mathcal{B})$. Denote $n = |E|$ and $E = \{e_1, e_2, \dots, e_n\}$. Also let r be the size of a base in $\mathcal{M}$. As in the algorithm given in Section 2, let us take a base B^* and without loss of generality denote $B^* = \{e_1, e_2, \dots, e_r\}$.

Then we define the root by B^* and we use 1-flip operations as before. The different is the parent function; we define the parent of $B \in \mathcal{B} \setminus \{B^*\}$ by $B - e_s + e_t$, where

- $t = \min\{i \mid e_i \in B^* \setminus B\}$
- $s = \max\{j \mid e_j \in B \setminus B^* \text{ and } B - e_j + e_t \in \mathcal{B}\}$.

(i) Prove (using (B2')) that the above parent function is well-defined and the parent-child relation forms a rooted-tree in the search graph.

(ii) Prove the following claim:

Claim 9. *Let B be a base, and let $e_k \in E \setminus B$. Also, let $C(B, e_k) = \{e_j \in B \mid B - e_j + e_k \in \mathcal{B}\}$. Then, for any $e_\ell \in B$, $B - e_\ell + e_k$ is a child of B if and only if (i) k is the maximum index among $C(B, e_k)$ and (ii) B contains e_i for all $1 \le i \le \ell$.*

$C(B, e_k)$ is called the *fundamental circuit* in matroid theory. Indeed it forms a cycle when $\mathcal{M}$ is a graphic matroid.

(iii) Sketch an implementation of the reverse search algorithm (using the above root and the parent function) that runs in $O(n(Q + r)|\mathcal{B}|)$ time, where Q denotes the (upper bound of) time to compute $C(B, e_k)$ for any $B \in \mathcal{B}$ and $e_k \in E \setminus B$.

References

[1] D. Avis and K. Fukuda. Reverse search for enumeration. *Discrete Applied Mathematics*, 65(1-3):21–46, March 1996.

[2] D. Avis, N. Katoh, M. Ohsaki, I. Streinu and S. Tanigawa. Enumerating constrained non-crossing minimally rigid frameworks. *Discrete and Computational Geometry*, 40(1):31–46, 2008.

[3] P. Bose and F. Hurtado. Flips in planar graphs. *Computational Geometry: Theory and Applications*, 42(1):60–80, 2009.

[4] M. Bern and D. Eppstein. Mesh generation and optimal triangulation. *Computing in Euclidean Geometry, 2nd Edition, Du and Hwang eds.*, 23–90, 1992.

[5] J. Oxley. *Matroid theory.* Oxford University Press, USA, 1992.

[6] N. Katoh and S. Tanigawa. Fast enumeration algorithms for non-crossing geometric graphs. *Discrete and Computational Geometry*, 42(3): 443–468, 2009.