Bounds On Some Geometric Transforms

by

Md. Kamrul Islam

A thesis submitted to the

School of Computing

in conformity with the requirements for

the degree of Master of Science

Queen's University

Kingston, Ontario, Canada

May 2005

Abstract

A *flip* or edge-replacement can be considered as a transformation by which one edge e of a geometric figure is removed and an edge f ($f \neq e$) is inserted such that the resulting figure belongs to the same class as the original figure. This thesis is concerned with transformation of two kinds of geometric objects, namely, planar trees and planar paths, through the application of flips. A technique is presented for transforming a given planar tree into another one for a set of n points in general position in the plane. It is proved that the number of flips required for such a transformation is at most $2n - k - s - 2$ ($k, s \geq 1$). In the case of planar path transformation we show that any planar path can be transformed into another by at most $2n - 5$ flips for a set of n points in convex position in the plane. Besides, experimental results are presented that show transformability of any planar path into another considering n ($n \leq 13$) points in general position. Later, we investigate the possibility of using flips, as an enumeration technique to generate the set $\mathcal{P}(S)$ of all planar paths of a set S of n points in convex position in the plane. First, it is shown that $|\mathcal{P}(S)| = n2^{n-3}$. Then two algorithms are proposed that describe the ways flips can be used to generate all the planar paths uniquely. The first algorithm is based on a recursion technique and uses flips of size one. The second algorithm is non-recursive and uses flips of size one and flips of size two (two edges are removed and two are inserted simultaneously) to enumerate all such paths uniquely.

Acknowledgements

First of all, I glorify the greatness and bounty of Allah who has bestowed on me the strength and ability without which it would not have been possible to carry out the thesis.

I am grateful to my supervisors Dr. Selim G. Akl and Dr. Henk Meijer who introduced me to the tremendously exciting field of research and guided me to the way of innovation and novelty. Their invaluable ideas and profound research experience kept me enthusiastic and optimistic all the way to the completion of this thesis. I thank them for their many hours of patience for listening to my problems. Their assistance, comments, constructive criticism and positive attitude helped me proceed towards completing each step of the thesis.

I would like to acknowledge the support and facilities I received from the staff of School of Computing and Queen's University.

I am indebted to my father Late Md. Anowarul Islam for introducing me the things I did not know and the utmost care I received from him. I pay my sincere gratitude and profound love to my mother, Mst. Fazilatun Nessa, who put up with hardships and kept patient to raise us (My brother, me and my sister).

Lastly, I thank my wife Mithila whose understanding, inspiration and constant support provided me with optimism in different situations.

Contents

List of Figures

List of Notations

P	set of n points in general position in the plane
S	set of n points in convex position in the plane
(v_i, v_j)	an edge connecting vertices v_i and v_j
$\mathrm{CP}_{i,i+1}$	canonical path begins with edge $(i, i+1)$
$\mathcal{T}(P)$	set of all planar trees of P
$T_G(P)$	graph with $\mathcal{T}(P)$ as vertex set
V_k^i	set of $k+1$ consecutive vertices $\{v_i, v_{i+1}, v_{i+2}, \cdots, v_{i+k}\}$
$N(v_j)$	the set of vertices which are adjacent to vertex v_j
$\mathcal{P}_k^i$	a set of paths of length k beginning at v_i and using V_k^j
p_k^i	number of directed paths of length k starting at v_i and using V_k^j
$T_{v_0}^*$	canonical tree with v_0 as the root
$\mathcal{P}(S)$	set of all planar trees of S
$\mathcal{P}_j(v_i)$	any path where j is the number of vertices visible to vertex v_i
$N_j(v_i)$	the number of child paths produced by $\mathcal{P}_j(v_i)$
$Qua(\mathcal{P}_{n-1}^i)$	quality of a path $\mathcal{P}_{n-1}^i$

Chapter 1

Introduction

Geometric elements such as points, line segments and polygons are the basis of a broad variety of different applications and give rise to an interesting set of well-defined problems and algorithms. One such geometric problem asks whether any geometric object or figure consisting of those geometric elements (such as straight line segments and fixed point-set in the plane) can be transformed into another object of the same type by applying small changes (e.g. removing an edge and replacing with another) in the object. If this question gives a positive answer then we would like to determine the minimum (maximum) number of such small changes. The constraint associated with such problem is that if there is a finite number of changes needed, then all the intermediate objects will be in the same class as the original one. Another interesting issue regarding transformation may be posed, namely, whether it is possible to generate the set of all objects uniquely with small changes in the objects. The process of generating the objects is called *enumeration*. In this thesis, we study the transformation of certain geometric objects and determine the bounds on those transformations.

There are a number of examples that illustrate this particular computational problem of transformation and enumeration. For instance, problems related to transformation and enumeration of geometric objects such as triangulations, tetrahedrons, spanning trees, crossing free paths, linked-edge lists, pseudo-triangulations and meshes are studied in the literature of computational geometry. Some of these algorithms, techniques and their lower and upper bounds to achieve those transformations and enumerations can be found in [3] [5] [29] [35] [36]. A majority of research activity in computational geometry is devoted to the design and analysis of algorithms and data structures, in an effort to find the fastest possible solutions to those problems.

1.1 Defining the Problem

As mentioned earlier, this thesis deals with the geometric problems that involve transformation of a given object defined on a set of points in the plane. The aim of the thesis is to describe techniques and bounds on the complexity of transforming geometric objects consisting of points and straight line segments. The problem can be generally defined as follows: Let X be a given geometric figure of a class with some property $\mathcal{M}$, and let Y be another object of the same class satisfying $\mathcal{M}$. We consider objects or figures that can be described in terms of vertices and straight line segments in the plane. It is intended to construct the figure Y by applying a sequence of small changes or transformations to X.

The main criterion is that when changes are applied on the figure X, each resulting object towards achieving Y from X, must posses the same property, $\mathcal{M}$. The first question is: Is it at all possible to achieve Y from X? Once the problem is solved as far as transformation is concerned then we figure out how many steps are required to do so and if not then what the lower and upper bounds on the number of such transformations are. One of the aims of this thesis is to answer these questions on particular geometric objects, namely planar trees and planar paths. We assume throughout that all problems are defined in the two-dimensional plane.

In the remainder of this section follows in this section, we shed light on what we mean by a local transformation in general. Subsequently we define and illustrate the flip operation as a local transformation which is one of the basic operations applied on geometric objects to produce other objects. Finally we present a particular example that will help one understand the transformation of geometric figures.

1.1.1 Geometric Transformation

In general, a *geometric transformation* can be defined as an operation in which new objects of the class are generated only from previously-examined objects by means of small changes. This transformation is performed on the vertices and the edges of a graph. As a new object is produced due to a single transformation of an object, the new object constitutes the neighbourhood of the original object. This simple transformation is performed successively on every newly produced object without any duplication of any objects. Thus we form a class of valid objects having similar characteristics.

We are concerned with the connectedness of all the objects of that class with respect to the transformation in question, that is, whether every valid object must be reachable from some initial valid object by means of a finite sequence of transformations. Local search (that searches only within the neighbourhood of the current object) and optimization methods such as reverse search enumeration, and random walks all make use of transformation to visit new objects of the class.

1.1.2 Flip as Geometric Transformation

The standard edge flip in a planar graph G, also called the Lawson flip [7], takes two triangles $\triangle abc$ and $\triangle abd$, whose union is a convex quadrilateral. By a flip we exchange the diagonals e and f of this quadrilateral and obtain a new graph G' as shown in Fig. 1.1.

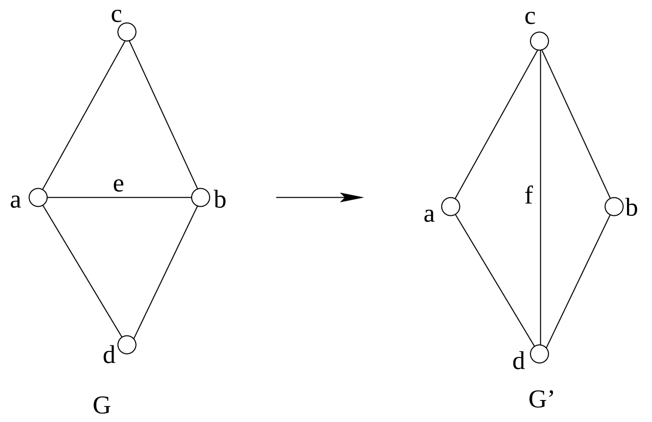


Fig. 1.1: Illustrating diagonal flips

We say that G' has been obtained from G by a diagonal transformation or a diagonal flip. Observe that one can flip only {a,b} if *adbc* is a convex quadrilateral. In this case it can be stated that the edge {a,b} is **flippable**.

Armed with the knowledge of the transformation flip, we can now look forward to considering the following example involving transformation of a geometric figure called a *triangulation*.

1.1.3 Triangulation Example

A triangulation is a an embedded planar graph all of whose faces (except the outer face) are triangles [30]. A set of points in the plane can be triangulated by connecting pairs of points with straight line-segments until no edges can be added without making an intersection with existing edges. Similarly the interior of a polygon can be triangulated by connecting its vertices with diagonals such that each diagonal lies entirely inside the polygon until no diagonal can be added without intersecting the existing diagonals.

Let X and Y be two triangulations with the same set of n vertices for $n = 7$ in the plane. It is required to obtain Y from X by applying a sequence of local transformations such that every resulting object still is a triangulation. Here we use flip as the local transformation of triangulations, which is defined earlier as the removal of one edge and the insertion of a new edge such that all the intermediate figures belong to the same class of the original figure. Fig 1.2 gives an example.

The questions associated with such transformations deal with whether it is always

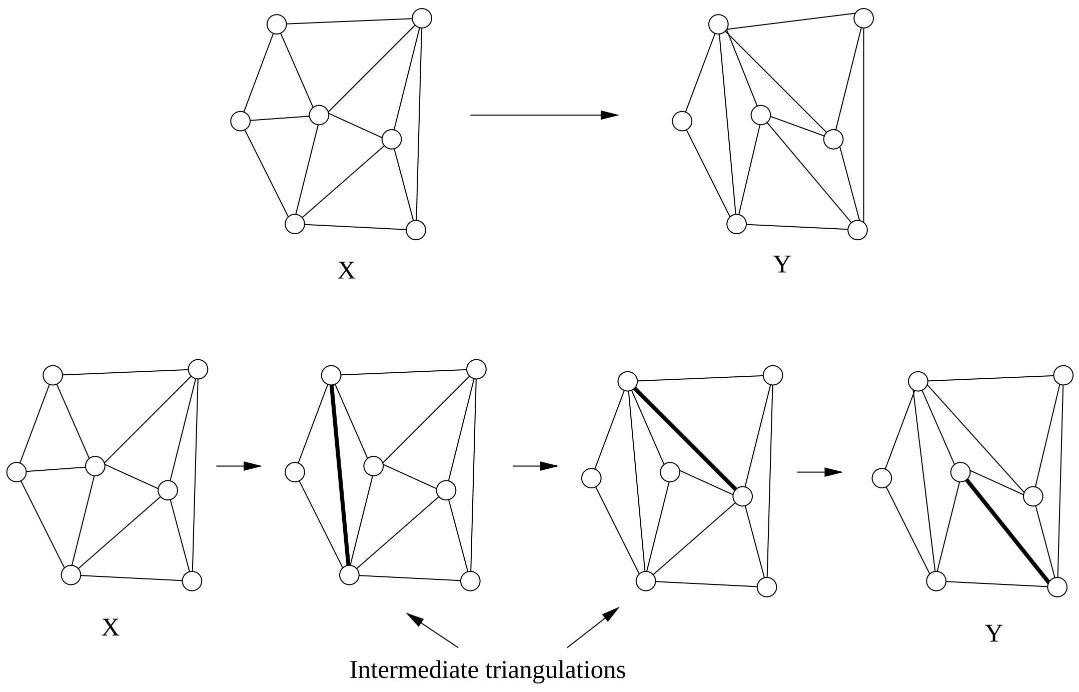


Fig. 1.2: Triangulation X is transformed into Y through flips.

possible to transform one object to another and if so then what is the number of such transformations. Both questions are settled with an affirmative answer and a lower bound is found to be $\Omega(n^2)$ [36] where n is the number of points in the plane.

1.2 Flips in Trees and Paths

In this thesis, we restrict ourselves only to the transformations of planar (sometimes referred as non-crossing or crossing-free) trees and planar paths. A planar tree is an embedded connected acyclic (containing no cycle) graph where the edges (straight line-segments) do not pairwise cross. Similarly a crossing-free or planar path is an embedded connected graph such that no two edges cross and every vertex has degree at most two.

The standard definition of a flip has been extended (flipping of edges is not restricted to only the diagonals of a quadrilateral) to general graphs such as planar trees and planar paths where a flip means the operation of removing an edge $e \in G$ and replacing it with another new edge f such that a new planar graph $G' = G\backslash\{e\} \cup \{f\}$ is formed. Here G and G' represent either planar trees or planar paths. See Fig 1.3(a) where a flip on the path $\mathcal{P}$ can transform it into any of the six paths $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$, $\mathcal{P}_5$, $\mathcal{P}_6$ as shown. But in Fig 1.3(b) the path $\mathcal{P}'$ produced from $\mathcal{P}$ by the flip is not valid as it is not planar.

A sequence of flips replaces a given set of edges with another set of the same cardinality. This operation (flip) can be used for the purposes of enumeration, random generation of geometric objects such as trees and paths (as we will see later) and for proving fundamental combinatorial properties of some objects of a certain class.

1.2.1 The Meta Graph and its Connectedness

One can then define a mathematical model in terms of transformations for all the objects of a class of whether any two geometric objects of a certain class (satisfying some property) are reachable from one to another via a finite sequence of flips. Here we define a graph, called the *meta graph* (also called *super graph*), that serves as the mathematical model to describe the relationships among the objects of that graph. The meta graph can be defined as the graph having the set of objects in the class as its vertex set, and a pair of vertices is connected by an edge if the objects represented

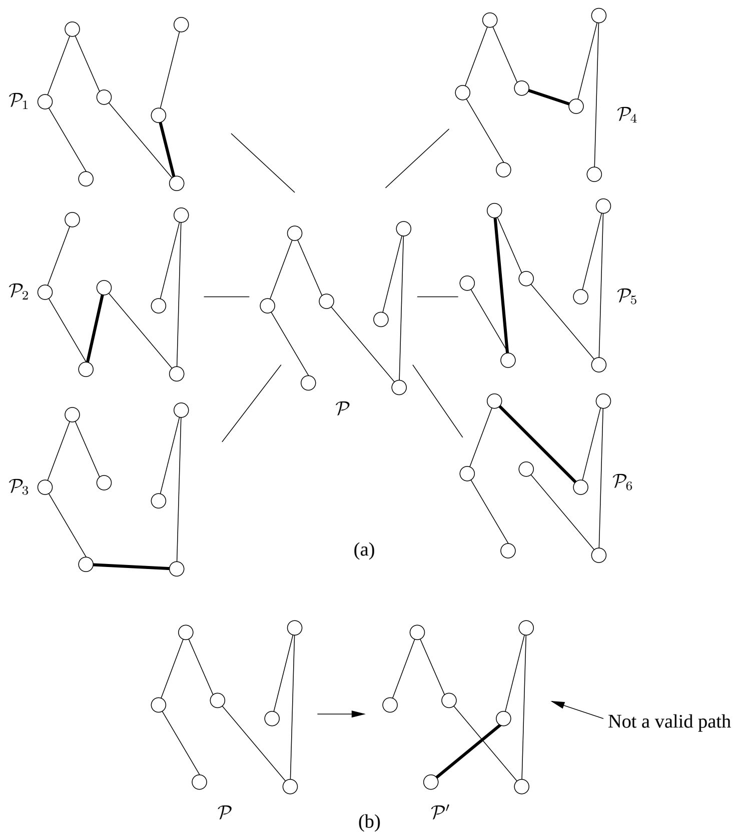


Fig. 1.3: (a) A flip on $\mathcal{P}$ can transform it into any of the six paths $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$, $\mathcal{P}_5$, $\mathcal{P}_6$. (b) The path $\mathcal{P}'$ produced from $\mathcal{P}$ by the flip is not valid as it is not planar.

by the vertices differ by a small change. In this way, one can develop adjacencies among nodes of the meta graph.

For a set of points in the plane in general position, the order of the meta graph is quite large. For example, the number of crossing-free trees is $\Omega(10.42^n)$ [25], the number of labelled non-crossing paths of n points in convex position in the plane is $n2^{n-3}$ (as will be shown later), and so on.

In most of the cases, generating, examining and establishing relationships among all the objects is not feasible due to the amount of time needed to construct a graph containing such a huge number of vertices. There is a way that makes it possible to establish polynomial-size descriptions among the objects even though the size of the graph is exponential in most of the cases. The procedure of incorporating local modifications on some initial object to visit new objects allows us to study the essential characteristics such as connectedness, reachability, diameter of the graph and so on. The existence of a path between two vertices in the graph means transformability of the corresponding trees or paths represented by the vertices into one another by means of repeated application of the local transformations. The length of the shortest path between two vertices corresponds to the distance between the two trees or paths in terms of the number of operations. Furthermore, analysis of such graph helps us develop suitable expressions to represent the relationship among the objects that arise from the issue of carrying out transformations.

In this thesis, we show that small changes in geometric figures of some class can be one of the means to enumerate all such figures (having same property) of the class. Besides, we aim to determine the bounds on the number of such transforms of planar trees and paths.

1.3 Enumeration of Objects

Enumeration entails the use of algorithms for generating all the combinatorial or geometric objects of a certain set without duplicates. The problem of efficiently generating the set of objects has theoretical interest and also is a fundamental problem in combinatorics, computational geometry and operations research. For example, generation of all graphs for a class may arise in constructing test data for certain programs. As this thesis is concerned with object transformations, we intend to extend this capacity to deal with enumeration of all geometric objects of a certain class. The techniques of generating or enumerating all planar paths are provided.

In order to find out efficient solutions regarding combinatorial objects for the purpose of classification and enumeration, numerous techniques have been investigated by mathematicians and geometers. One of the earliest object classes to be studied by the ancients is likely the Platonic solids [16], for example, a polyhedron all of whose faces are congruent regular polygons. Along with theoretical solutions and proof of enumeration techniques, computers have also been employed to obtain experimental results that investigate the structures of objects far more complex than those considered by the ancients and help provide certain proofs. For example, in graph theory, the only proofs yet known of the famous Four Colour Theorem have been obtained by means of computer enumeration of cases [1][2].

Since the size of the meta graph (as described above) is quite large, it takes a lot of time to enumerate all objects represented by the vertices of the meta graph. Nonetheless, for the sake of determining connectedness of the meta graph whose vertices represent planar paths, when the vertices are in general position (no three points are collinear), we apply a brute-force enumeration algorithm and take advantage of the computer to enumerate all such paths of n vertices for $n \leq 13$ when the vertices are in general position in the plane. Later, two theoretical proofs are given for the enumeration of all planar paths of n vertices assuming that the vertices are in convex position.

1.4 Contribution of the Thesis

As we deal with transformation of geometric objects, we determine the minimum and maximum number of flips required for such transformation which are called the *lower and upper bounds of transformation.* The problem of finding bounds of transformation of crossing-free spanning trees and planar paths is addressed and some algorithms are suggested. Besides, some results are given by studying the combinatorial and enumerative property namely, the counting of the total number of objects of a certain class, generating them all, finding relationship between any two successively generated objects, and so on. Following are the contributions of this thesis:

1. It is shown that at most $2n-5$ flips are required to transform any crossing-free path into another when the point set is in convex position.

2. The number of crossing-free paths in convex position is shown to be $n2^{n-3}$. Two enumeration algorithms (one recursive and another non-recursive) are designed for generating all such paths.

3. The non-recursive enumeration algorithm for generating paths in convex position is implemented. Although the time complexity is exponential, the space complexity is linear.

4. An algorithm is presented for the problem of planar tree transformation and it is shown that a linear number of flips suffice to transform any planar tree into another planar tree considering points in general position in the plane. More specifically, we improve the upper bound on the number of transformations from $2n - 4$ [3] to $2n - k - s - 2$ where both k and s are larger than or equal to 1.

5. Experimental results concerning transformation of crossing-free paths are presented for at most $n = 13$ points in general position in plane. It is shown that any two paths are reachable from one another which implies that the corresponding meta graph is connected.

1.5 Outline of the Thesis

The rest of the thesis is organized as follows. Chapter 2 provides a brief literature survey on flips that act as the basis of transforming triangulations and general graphs and sheds light on some of its applications. In chapter 3 some basic definitions and terminologies necessary for understanding the thesis are provided. In chapter 4, a number of lemmas and proofs are provided regarding planar trees and crossing-free paths transformation. Experimental results demonstrating the connectedness of the meta graph containing vertices representing non-crossing planar paths are presented in this chapter. In chapter 5, a simple recursive technique for counting the total number of planar paths is given. Two algorithms are discussed for enumeration of all convex planar paths uniquely. The first one is recursive, where it is shown that flipping can be used to generate all planar paths without duplicates. We then present a non-recursive algorithm that takes only $O(n)$ space to enumerate all such paths. Chapter 6 provides a summary of the results and concludes with an open problem.

Chapter 2

Literature Review

2.1 Flips and Triangulations

The idea of flipping in planar tree and path transformations stems from the edge-flipping operations used in triangulations, a structure which has great importance in computational geometry. Flipping can be used in generating different types of triangulations in two and three dimensions [36][35], enumerating rooted plane triangulation [12] and also in building visibility graphs of a set of objects in the plane [34]. Flipping of edges as a simple tool of changing certain geometric objects, has received some attention during recent years. Flipping has found its applications in many important areas such as finite element analysis, solid modeling, shape representation, terrain modeling, volume rendering and computer vision [17][18][19].

Flips have been used in various fields starting with a simple greedy algorithm that constructs the *Delaunay* triangulation of a point set by successive flips from an arbitrary initial triangulation of the point set [8]. The results of [12] provide some efficient ways of enumeration and a number of algorithms for the generation of all tri-connected *rooted* plane triangulation defined with n vertices. The latter are of particular importance as they are based on the reverse search method that does not store all the lists of graphs while they are generated successively.

2.2 Flips in Trees

Apart from the literature regarding flips and their influence on triangulations, a considerable amount of similar study has been also conducted for general graphs. The meta graph of trees (as discussed in chapter 1) was introduced in [24], in connection with the study of electrical networks. An characteristic of the meta graph, specifically that the graph of trees is Hamiltonian is proved in [24]. A simpler proof of the same fact was found later [21] and generalized to the base graph of a matroid. A base graph of a matroid is the graph whose points are the bases of the matroid. Two bases are adjacent if they differ by exactly one element. Graph-theoretical versions of the problem of geometric object transformations have been largely studied in [22] for tree graphs and was shown that tree graphs have maximum connectivity (a directed graph is said to be maximally connected if it is k-connected and k is the minimum in or out degree of all vertices). Another excellent discussion of combinatorial version for graphs more general than trees is studied in [23]. They exclusively treat the case where the allowed operation is called *edge-move*, which relates two trees in the meta graph of trees if they have all but one edge in common.

Motivated by the question of enumerating the set of all non-crossing spanning trees for a point set in general position, Avis and Fukuda [3] showed that the corresponding tree graph is connected and has a diameter bounded by $2n - 4$. To the best of our knowledge this is the only known result for a general point set. In this thesis, we present an improved bound of $2n - k - s - 2$ (where $k, s \geq 1$) which in some cases produces a better result.

The special case when the point set is in convex position has been treated by Hernando et al. [26]. They showed that the tree graph has maximum connectivity and is Hamiltonian in this case. A lower bound of $\lfloor 3n/2 \rfloor - 5$ on its diameter is established there and it is shown that the number of crossing-free spanning trees is minimum for the convex case. A different flavour on crossing-free geometric spanning trees and related results can be found in [27].

2.3 Edge Slide and Improving Edge Move

Recently, two operations slightly different from but related to flips have gained attention in connection with transformations; they are called *edge-slide* and *improving edge-move* [28]. A planar edge-slide is like a constant-size local edge move that keeps one endpoint of the moved edge fixed and moves the other one along an adjacent tree edge without generating any edge crossings. An improving edge-move is same as the edge move or flip, the only difference being that it (move) reduces the total tree length. The length of a tree is the sum of lengths of all the edges of the tree.

As found in [28], the number of length-improving and planar edge-moves needed to transform a tree $T \in T_s$ into $MST(S)$, (Minimum Spanning Tree) of S where T_s is the set of all crossing-free spanning trees of S, is $O(n \log n)$. Edge slide operations could also prove useful in enumerating all simple polygons on a point set S via constant-size local transformations but this is still unsettled as reported in [26].

2.4 Enumeration Techniques

There are techniques available for enumeration of typical objects such as generation of all rooted trees [12], triangulations of a set of points in the plane [37], non-crossing paths in a connected graph in a two-dimensional Euclidean plane, all connected induced subgraphs of a graph, all topological orderings of an acyclic graph, vertices and faces of a polyhedron etc [3]. As described in [11], one of the particular applications of enumerating all graphs having some property includes unbiased statistical analysis. Davis and Fukuda [14] provide a nice treatment regarding algorithms developed for counting and generating these objects. As enumeration techniques are in place, the necessity of searching a particular object also demands attention. Among the number of searching techniques backtracking is known to be useful for various enumeration problems associated with graphs [20]. For enumeration problems in computational geometry the incremental search technique is used [13]. On the other hand two most typical search techniques in graphs namely, depth first search and breadth first search can be applied to the case where the objects to be listed or counted are the vertices of some connected graph.

The experimental results presented in this thesis for finding transformability of planar paths use flips to sequentially traverse all the planar paths of a point set in general position. That is, we first generate all the planar paths of the point set. Then to answer the question as to whether any two planar paths P_1 and P_2 can be reached from one another by means of flips, the algorithm begins from P_1 and uses a flip to reach its neighbour which in turn reaches to its neighbour and so on until P_2 is reached. This can be called local search as the search is limited only to the neighbours of a path represented by a vertex of the meta graph. There are local search techniques discussed in the literature [8][9] of enumeration and searching algorithms. For example, edge-exchange algorithms for finding a particular object with some property in some weighted graph and flip algorithm for finding a Delaunay triangulation in the plane are studied in [8][9][10][13].

Chapter 3

Preliminaries and Definitions

A number of definitions are provided in this section. These definitions and notations will be used throughout the rest of the thesis. In order to follow our discussion, the following are some basic definitions and concepts given without proof. Some of them are adopted from [31][32][33].

3.1 Graph Essentials

Graph: A **graph** $G = (V, E)$ consists of V, a nonempty set of vertices, and E, a set of edges between the vertices, $E = \{(v_i, v_j)|v_i, v_j \in V\}$. If G is drawn in the plane, an edge (v_i, v_j) is represented by a straight line-segment joining two vertices v_i and v_j. The cardinalities of V and E are denoted by $|V|$ and $|E|$ respectively. A **path** from v_i to v_j is a sequence of edges where the terminal vertex of an edge is the same as the initial vertex in the next edge in the path. A **cycle** in G is a path of length

($n \geq 3$) that begins and ends at the same vertex. A graph is called **acyclic** if it contains no cycles. A graph is called **connected** if for every pair v_i, v_j of distinct vertices there is a path from v_i to v_j. A graph G is called **planar** if it can be drawn in the plane so that no two edges intersect, except at a common vertex. We also call such a drawing an embedding of graph G. Throughout the rest of the thesis a graph G drawn in the plane is assumed to be a planar embedding unless otherwise stated. If $(v_i, v_j) \in E$, then v_i and v_j are **adjacent** or **neighbours**, and the edge (v_i, v_j) is said to be **incident** to v_i and v_j. We define $N(v_j)$ to denote the set of vertices which are adjacent to vertex v_j. The **degree** of a vertex v_i is the total number of edges incident to it. It is denoted by $deg(v_i)$. A **flip** in G is the operation of removing an edge from G and adding an edge to G. If k ($k \geq 1$) edges are removed from, and k are added to, G then we call this operation a flip of size k.

3.2 Terminologies used for Trees

In the following definitions, assume $P = \{v_0, v_1, v_2, \cdots, v_{n-1}\}$ is a set of n points in general position (recall that general position means no three points are collinear) in the two-dimensional Euclidean plane. We assume that trees are drawn in the plane. Vertices (V) and edges (E) of a tree are represented by points of P and straight line-segments.

Planar tree: A **planar tree** $T = (V, E)$ is a connected acyclic planar graph.

Rooted tree: A **rooted tree** is a planar tree in which one vertex is designated as the **root** of the tree.

In this thesis, it is considered that the **root** of a tree $T = (V, E)$ is the vertex with minimal $x-$coordinate. The vertex with smallest $y-$coordinate will be designated as the root if there are two or more candidates for the root. The root of any tree, T will be represented by v_0.

Canonical tree: A **canonical tree** $T^*_{v_0}$ is a rooted tree where all vertices are adjacent to v_0, that is, in $T^*_{v_0}$, $v_i \in N(v_0), \forall_i, i \neq 0$. Fig. 3.1 shows the canonical tree and a tree which is not canonical.

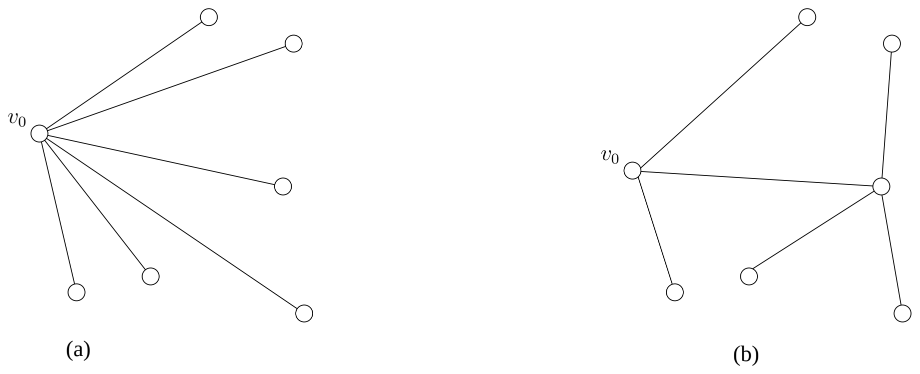


Fig. 3.1: (a) The canonical tree (b) Not a canonical tree

Visibility of vertices: Two vertices v_i and v_j, $v_i \neq v_j$ in an embedding of G are **visible** to each other if the straight line segment $(v_i, v_j) \in E$ between them does not intersect any of the edges in G.

It is assumed that v_i is not visible to v_j if $v_i \in N(v_j)$. If v_i and v_j are not visible then they are blocked by some edge in G. This is illustrated in Fig. 3.2.

Let $\mathcal{T}(P)$ denote the set of all trees of P and the geometric tree graph $T_G(P)$

denote the graph having $\mathcal{T}(P)$ as vertex set. Two trees $T_1, T_2 \in \mathcal{T}(P)$ are adjacent if $T_2 = T_1 \backslash \{e\} \cup \{f\}$ for some edges e and f.

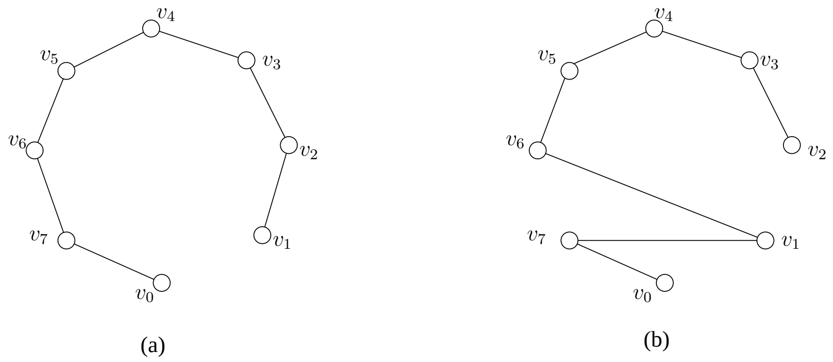


Fig. 3.2: In (a) v_1 can see vertices $v_3, v_4, v_5, \cdots, v_0$ and in (b) v_2 can see only v_4, v_5, v_6 and v_1

In the rest of the thesis, it is assumed that a tree is planar unless otherwise mentioned.

Flip in a tree: A **flip** in a tree T_1 is the operation of removal an edge e and addition of a edge f so that $T_2 = T_1 \backslash \{e\} \cup \{f\}$ is a tree.

3.3 Terminologies used for Paths

In the following definitions, assume a point set $S = \{v_0, v_1, v_2, \cdots, v_{n-1}\}$ is a set of n points in convex position in the two-dimensional Euclidean plane. Assume that no two points of S have the same $y - coordinate$s otherwise we rotate the points accordingly.

Denote the vertex with the lowest $y - coordinate$ in S as v_0 and label the rest of the vertices $v_1, v_2, \cdots, v_{n-2}, v_{n-1}$ counter clock-wise from v_0. Arithmetic on the indices of vertices of paths will be defined modulo n. In the rest of the thesis, it is assumed that all paths are drawn in the plane whose vertices (V) and edges (E) are represented by points of S (unless a different point-set is specified) and straight line-segments.

Consecutive group of vertices: A set of $k+1$ vertices, $v_i, v_{i+1}, v_{i+2}, \cdots, v_{i+k-1}, v_{i+k}$ is called a **consecutive group of vertices** and denoted by V_k^i.

Let $\mathcal{P}(S)$ denote the set of all planar paths on S. Henceforth, whenever we mention a path we mean it is a planar path.

Let $\mathcal{P}_k^i$ denote any path of k edges beginning at v_i and using all vertices of V_k^j for some j such that $v_i \in V_k^j$. Note that a path $\mathcal{P}_k^i$ is a Hamiltonian planar path if $k = n - 1$.

Also, let p_k^i define the number of directed paths of length k starting at v_i and connecting vertices of V_k^j for some j such that $v_i \in V_k^j$.

End-edge of a path: The edge (v_i, v_j) of a path $\mathcal{P} \in \mathcal{P}(S)$ is called an **end-edge** of $\mathcal{P}$ if v_i or v_j has degree one.

Canonical path: A directed path $(v_i v_{i+1} v_{i+2} \cdots v_{i-2} v_{i-1})$ (resp. $(v_i v_{i-1} v_{i-2} \cdots v_{i+2} v_{i+1})$) in $\mathcal{P}_{n-1}^i$ is called a **canonical path** and denoted by $\mathrm{CP}_{i,i+1}$ (resp. $\mathrm{CP}_{i,i-1}$).

The two end-edges of $\mathrm{CP}_{i,i+1}$ are (v_i, v_{i+1}) and (v_{i-2}, v_{i-1}). The path $\mathrm{CP}_{i,i+1}$ can also be represented as $\mathrm{CP}_{i-1,i-2}$.

Fig. 3.3 shows an example of a canonical path and a path that is not canonical.

Furthest (Nearest) Visible Vertices: A vertex v_j is called **furthest (nearest)** visible vertex from v_i with respect to a path $\mathcal{P} \in \mathcal{P}(S)$ if v_j is visible to v_i and the

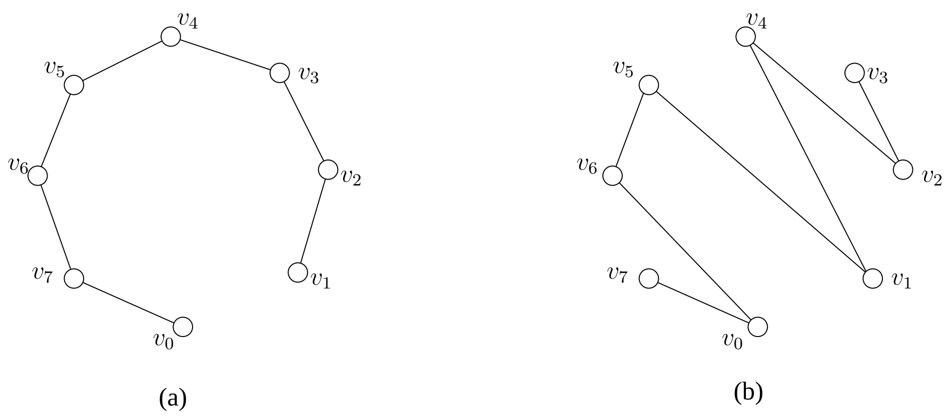


Fig. 3.3: (a) Canonical path $\mathrm{CP}_{1,2}$ and (b) not a canonical path.

number of edges walking along the path from v_i to v_j is maximum (minimum).

It is clear from the context that Furthest (Nearest) visible vertices will always be considered with respect to a path $\mathcal{P}$ whether mentioned or not.

Flip in a path: A **flip** in a path $\mathcal{P}_1$ is the operation of removal of an edge e and addition of a edge f so that $\mathcal{P}_2 = \mathcal{P}_1 \backslash \{e\} \cup \{f\}$ is a path.

Path generation: A path $\mathcal{Q}_k^j$ is called **generated** from $\mathcal{P}_k^i$ if a flip transforms $\mathcal{P}_k^i$ into $\mathcal{Q}_k^j$.

We say that path $\mathcal{Q}_k^j$ is generated from $\mathcal{P}_k^i$ by a single flip.

Quality of a path: The **quality** of a path $\mathcal{P}_{n-1}^i$ is defined to be the number of edges on the boundary of the convex hull of S and is denoted by $Qua(\mathcal{P}_{n-1}^i)$.

It can be verified that $Qua(\mathcal{P}_{n-1}^i) \geq 2$.

Chapter 4

Transformation Results

In this chapter, we discuss flipping operations for transformation of spanning trees and paths . The focus is mainly centered on finding the bounds on the number of flips required to transform trees and paths. In sections 4.1 and 4.2 we provide the technique and find the bound in terms of the number of flips used in such transformation. In section 4.3, our path transformation strategy is explained along with the bound needed to achieve the transformation.

4.1 Tree Transformation

Let $T' = (V, E')$ and $T'' = (V, E'')$ be any two trees belonging to $\mathcal{T}(P)$ (recall that $\mathcal{T}(P)$ denote the set of all trees of point set P). It is required to construct T'' by applying a sequence of flips one by one to T'. The number of flips required for such construction is known to be at most $2n - 4$ [3]. In general we say that T'' can be transformed from T' by p flips if there is a set of trees $T_0, T_1, \cdots, T_p$ where $T' = T_0$ and $T'' = T_p$ such that T_{t+1} can be obtained from T_t by a single flip. This implies that for any t, T_t and T_{t+1} are adjacent in $T_G(P)$.

Consider the Fig. 4.1 where the tree T'' is obtained from T' by a sequence of transformation.

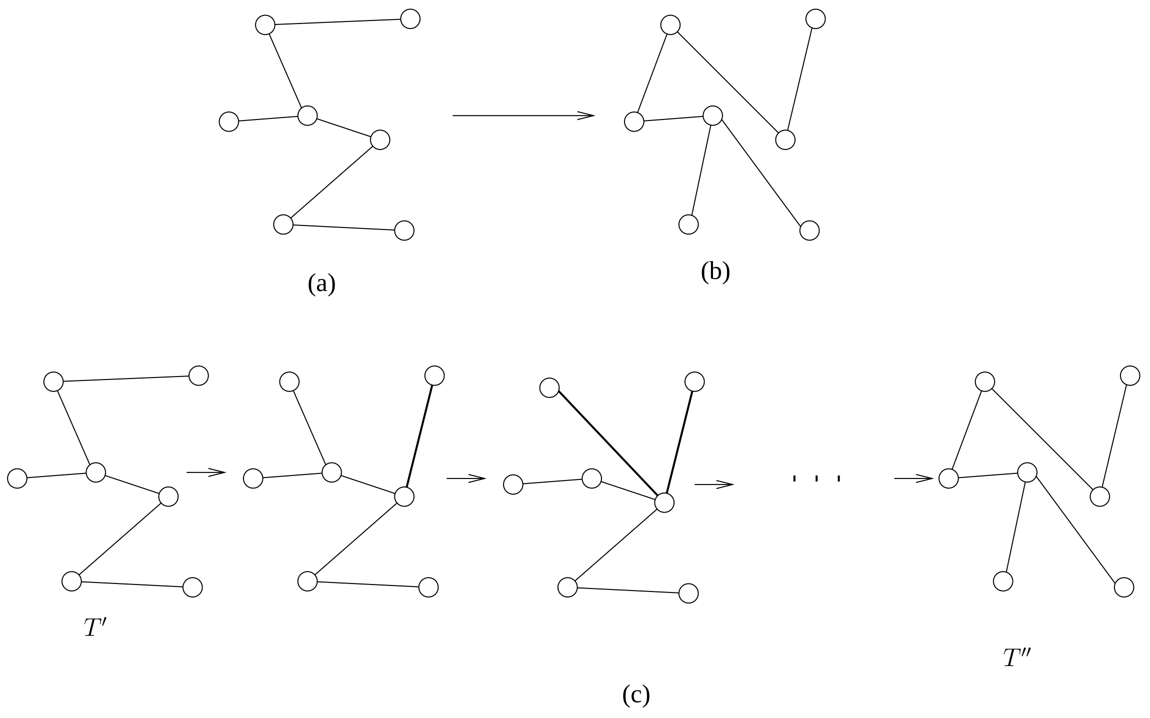


Fig. 4.1: (a) A tree $T' = (V, E')$ (b) A tree $T'' = (V, E'')$ (c) Transformations (shown with thick edges) applied on T' to construct T''.

In this thesis, we try to improve the upper bound $(2n-4)$. In the following section it is proved that two trees $T', T'' \in \mathcal{T}(P)$ can be transformed to each other with at most $2n-k-s-2$ flips, where $k > 0$ and $s > 0$ are the number of neighbours of the roots of T' and T'' respectively.

Instead of proving directly that a tree can be transformed into another tree the strategy is to show that it is always possible to transform the given tree into a unique canonical tree. One can then perform the reverse operations that transform the target tree into the same canonical tree.

4.1.1 Transformation via Canonical Tree

The following lemma provides a useful direction towards proving the main result:

Lemma 4.1.1 *At least one vertex of $T \in \mathcal{T}(P)$ and $T \neq T^*_{v_0}$ is visible to the root, v_0 of T.*

Proof Since the tree $T = (V, E)$ is not in its canonical form, $\exists i$ such that $v_i \notin N(v_0)$. Let N_1 and N_2 be the set of neighbours of v_0 such that $deg(v_p) = 1$ for $v_p \in N_1$ and $deg(v_q) > 1$ for $v_q \in N_2$ with $0 < p, q \leq n-1$. According to the definition, v_0 can see neither the vertices of N_1 nor those of N_2.

Let ℓ_1 be a line segment connecting v_0 to any of the vertices $v_r \notin N(v_0)$. Denote by $\angle\ell_1$ the angle made by ℓ_1 with v_0 along the vertical line that goes up through v_0. If ℓ_1 does not cross any edge of E, this means v_r is visible to v_0 and the proof is done. So we may assume that ℓ_1 crosses some edges of T. Denote the set of intersected edges E_{int_1}.

Select the edge $e_1 \in E_{int_1}$ whose intersection with ℓ_1 is nearest to v_0.

Now at least one of the vertices of $e_1 = (v_1, v_2)$ must not be a neighbour of v_0, otherwise if both of them are neighbours of v_0 then it is a loop. Assume v_1,$v_2 \notin N(v_0)$. In this case we can arbitrarily choose v_1 or v_2 to connect to v_0. If one of them is a neighbour then we select the other vertex which is not a neighbour of v_0 and connect it to v_0. Let ℓ_2 denote a line segment joining v_0 to one of the vertices v_1 or v_2 . Now check whether the edge ℓ_2 crosses any edges of T. If there is no crossing then the other vertex on ℓ_2 is visible to v_0. If this is not the case, i.e., $|E_{int_2}| > 0$, then choose the closest intersected edge e_2 from E_{int_2} from the root and follow the above procedure of connecting v_0 to the vertex of e_2 which is not neighbour of v_0. Repeatedly applying the procedure described above gives rise to the following cases:

Case 1: Case 1 deals with the situation where $\angle\ell_1 < \angle\ell_2 < \angle\ell_3 \cdots < \angle\ell_a$ (here $a \leq n-2$). Since there is a finite number of vertices in the graph, by following the above procedure we can reach to a vertex v_s where the edge ℓ_a (one end vertex is v_s and the other one is v_0) will not cross any edges. This ensures us that in the worst case ($a = n-2$) the line segment ℓ_a makes the largest angle joining the vertex v_s to v_0. Then v_s is the vertex visible to v_0. This is illustrated in Fig. 4.2.

Case 2: Case 2 is the reverse of case 1 ($\angle\ell_1 > \angle\ell_2 > \angle\ell_3 \cdots > \angle\ell_a$) and can be similarly resolved.

Case 3: In Case 3 we consider the following:

There exists some i such that (i) $\angle\ell_i < \angle\ell_{i+1} > \angle\ell_{i+2}$ or (ii) $\angle\ell_i > \angle\ell_{i+1} < \angle\ell_{i+2}$

Without loss of generality consider (i) $\angle\ell_i < \angle\ell_{i+1} > \angle\ell_{i+2}$. Now the line segment ℓ_{i+2} must lie between ℓ_{i+1} and ℓ_i and e_i. Let v_i and v_{i+1} be the end vertices of the line segments ℓ_i and ℓ_{i+1} respectively. Any ℓ_j (for $j > i+2$) must lie in triangle formed by the line segments ℓ_i,ℓ_{i+1} and e_i because at each step we are taking the closest intersected edges to v_0. Also this implies that at every iteration the number

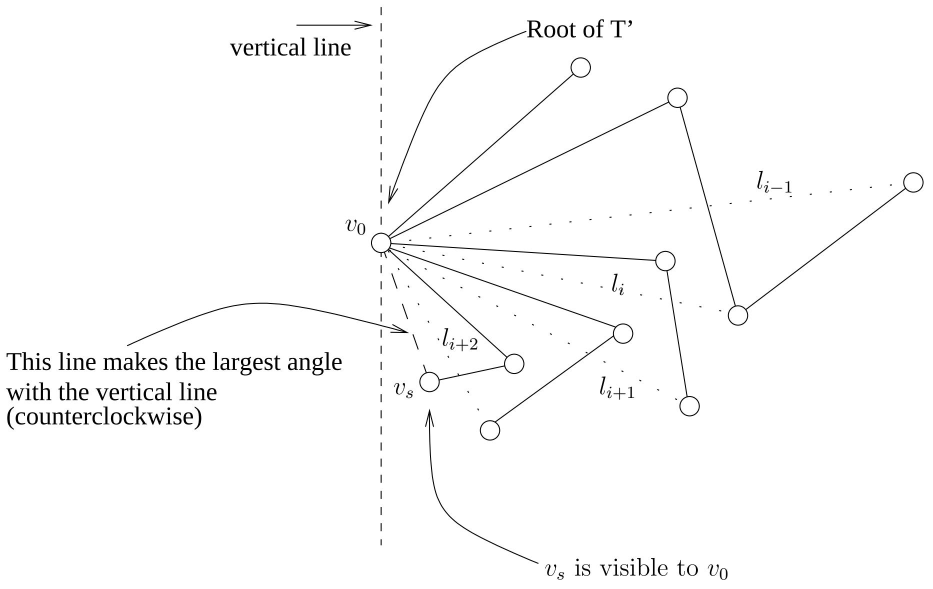

vertical line
Root of T'
v_0
l_{i-1}
l_i
l_{i+2}
l_{i+1}
v_s
This line makes the largest angle with the vertical line (counterclockwise)
v_s is visible to v_0

Fig. 4.2: A tree T of case 1. Only the solid line segments represent the edges of the tree.

of vertices to be considered for visibility to v_0 are reduced (only those vertices inside the triangles) as the area of successive triangles are decreased gradually. Since the number of vertices are finite and the number is becoming smaller and smaller at each step, we must end up with a single vertex v_s inside the smallest of such triangles (in the worst case) that will be visible to v_0. Fig. 4.3. shows such a scenario. ∎

Lemma 4.1.2 *Any tree $T \in \mathcal{T}(P)$ and $T \neq T^*_{v_0}$ can be transformed into its canonical form $T^*_{v_0}$ through flipping of edges at most $(n-k-1)$ steps, where $k = |N(v_0)|$.*

Proof Begin with any vertex visible to v_0 (according to Lemma 4.1.1 at least one vertex of T will be visible to the root). Assume v_0 can see v_j. Since adding a line segment (v_0, v_j) makes a unique cycle $v_0 \cdots v_i v_j v_0$ in T, break the cycle by eliminating the edge (v_i, v_j). This flip increases the degree of v_0 by one. Then continue the same

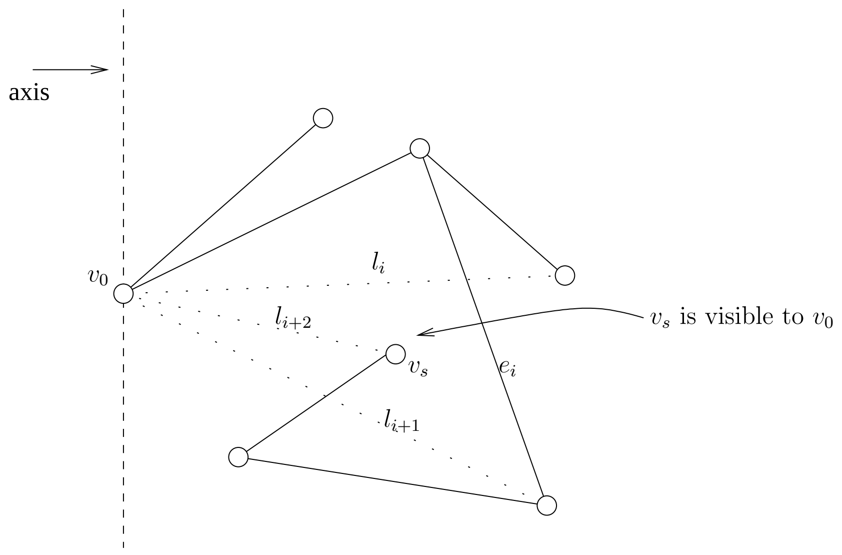


Fig. 4.3: A tree T depicting case 3. Only the solid line segments represent the edges of the tree.

process with another vertex visible to v_0 until $\forall v_i \in N(v_0)$ and $v_i \neq v_0$. An illustration is Fig. 4.4 shows the transformation of T into canonical tree $T^*_{v_0}$ through successive flips.

Since for any tree there can be $n-1-k$ edges possible which are not incident to v_0, the total number of flips required to produce $T^*_{v_0}$ is $n-1-k$. ∎

Example Given the tree in Fig. 4.4 with $n = 10$ and $k = 1$ note that the number of flips to obtain $T^*_{v_0}$ from the initial tree T is $n - k - 1 = 10 - 1 - 1 = 8$.

We will prove one of the main results of this thesis, i.e., two trees $T', T'' \in \mathcal{T}(P)$ can be transformed into one another with only a linear number of flips. Instead of directly proving that a tree can be transformed into another tree our strategy is that we transform T' into the unique canonical tree $T^*_{v_0}$. And then we can reverse the operations that transform T'' into $T^*_{v_0}$. This is shown in the following lemma:

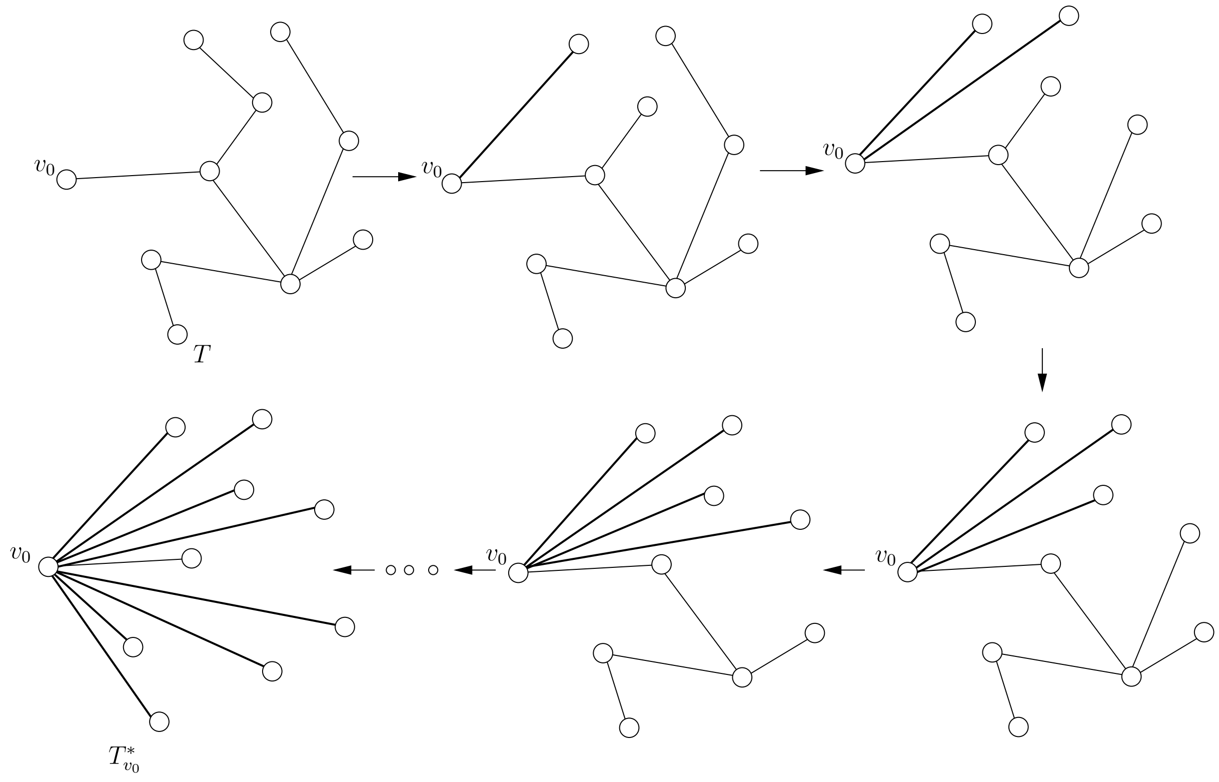


Fig. 4.4: Showing the transformation of tree T into canonical tree $T^*_{v_0}$. Thick edges represent the edges that are flipped.

Lemma 4.1.3 *Two trees $T', T'' \in \mathcal{T}(P)$ can be transformed to each other with at most $2n - k - s - 2$ flips where $k \geq 1$ and $s \geq 1$ are the number of neighbours of the roots of T' and T'' respectively.*

Proof Given the tree T', one can transform it into canonical tree $T^*_{v_0}$ using Lemma 4.1.2 with at most $n-k-1$ steps where k is the number of neighbours of v_0. Similarly, the tree T'' can be transformed into canonical tree $T^*_{v_0}$ with at most $n - s - 1$ flips where s is the number of neighbours of T''. Since the canonical trees for T' and T'' are same, it takes $2n - k - s - 2$ flips to transform T' into T''. This completes the proof. ▮

The following corollary follows from the above lemma.

Corollary 4.1.4 *The number of flips required to transform a tree T' into another tree T'', where $T', T'' \in \mathcal{T}(P)$ is $2n - p - 2$ where $p = max_{v \in CH(V)}$(degree v in T' + degree v in T'').*

Suppose that T' and T'' are two trees with $n = 7$ as shown in Fig. 4.5. The neighbours of the roots of T' and T'' are 3 and 2 (hence $k = 3$ and $s = 2$) respectively. It takes 3 flips for T' to be modified to $T^*_{v_0}$ and 4 flips for T'' to be transformed to $T^*_{v_0}$. Hence we need $2n - k - s - 2 = 14 - 3 - 2 - 2 = 7$ flips to transform T' to T''.

The above result can be expressed in terms of the meta graph $T_G(P)$ whose vertices are the non-crossing set of trees, $\mathcal{T}(P)$. Two vertices of the meta graph are adjacent if the trees represented by them can be obtained from the other by a single edge replacement. According to the proof of the above lemma, we can derive that $T_G(P)$ is connected and the diameter of $T_G(P)$ which is the shortest distance between any two furthest vertices, can be at most $2n - k - s - 2$. As mentioned earlier, Avis and Fukuda [3] showed that the corresponding tree graph has a diameter bounded by $2n - 4$. We improve their result by $2n - k - s - 2$ which produces a better result when the values for k or s are greater than one. Even in the worst case when both the parameters k and s have the value equal to one our result is as good as theirs.

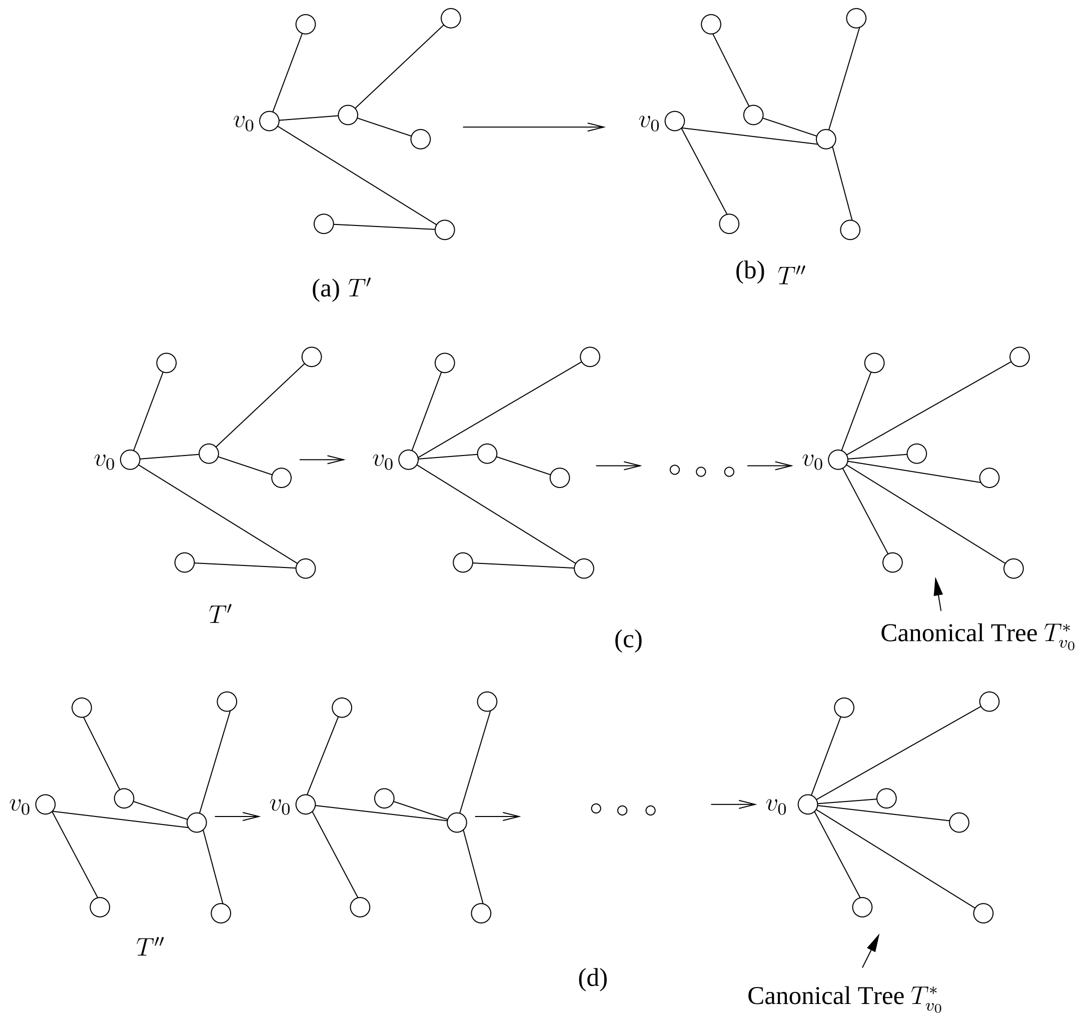


Fig. 4.5: Showing the transformation of tree T' and T'' to be transformed into canonical tree $T^*_{v_0}$.

4.2 Path Transformation

In this section both theoretical and experimental results regarding path transformation are presented. We were not able to prove that a path can be transformed into another when the point set is in general position in the plane. Instead of tackling the problem of path transformation with points in general position (as we did for tree transformation in the previous section) we conducted a number of experiments. We generated n random points ($n \leq 13$, and points are in general position and no three points are collinear) in the plane and all the planar paths were constructed. Then brute force search was employed to traverse all the paths of the point set and it was found that the path graph (defined earlier) was connected in all of the simulation runs. Section 4.2.3 provides the details of the experiment.

In section 4.2.2 we consider a special point set (points in convex position) and prove the theoretical result that any two paths can be transformed from one to another. It is shown that it takes a linear number of flips to transform a path into another path.

Assume we have the following information regarding the path graph. Define the geometric path graph as the graph having $\mathcal{P}(S)$ as vertex set and two paths $\mathcal{P}_k^i, \mathcal{Q}_k^j \in \mathcal{P}(S)$ are adjacent if $\mathcal{Q}_k^j = \mathcal{P}_k^i \backslash \{e\} \cup \{f\}$ for some edges e and f. In the following section, it is proved that two arbitrary paths in $\mathcal{P}(S)$ can be transformed to each other with at most $2n-5$ flips if $n = |S|$.

4.2.1 Quality of a Canonical path

Recall that a canonical path $\mathrm{CP}_{i,i+1}$ on S (recall that S is set of points in convex position) is a path where the edges (v_i, v_{i+1}) are defined on the boundary of the convex hull of S. This means that the quality of $\mathrm{CP}_{i,i+1}$ is, $Qua(\mathrm{CP}_{i,i+1}) = n-1$ (as defined earlier the *quality* of a path is the number of edges that are on the boundary of S), see the Fig. 4.6.

The quality of any path is at least two because a path P which starts at v_i must

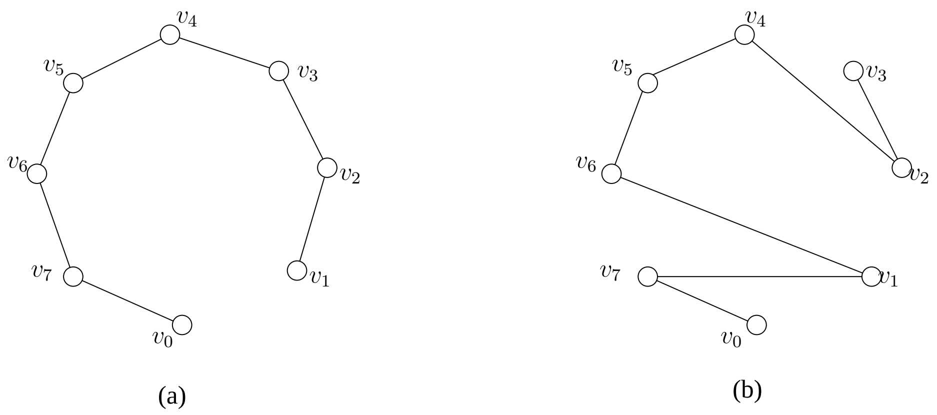


Fig. 4.6: (a) Shows a canonical path, $\mathrm{CP}_{0,1}$ with $Qua(\mathrm{CP}_{i,i+1}) = 7$ and (b) shows an arbitrary path $\mathcal{P}$ with $Qua(\mathcal{P}) = 4$.

have the starting edge v_iv_{i+1} or v_iv_{i-1} which is on the boundary of the convex hull. This is also true for the other end of P. So the quality of any path P is at least two.

4.2.2 Path Tranformation Strategy

Like the canonical transformation for trees, instead of directly transforming a path $\mathcal{P}^a_{n-1} \in \mathcal{P}(S)$ into another path $\mathcal{P}^b_{n-1} \in \mathcal{P}(S)$ where $\mathcal{P}^a_{n-1} \neq \mathcal{P}^b_{n-1}$, we first show that it is always possible to transform $\mathcal{P}^a_{n-1}$ into a canonical form $\mathrm{CP}_{i,i+1}$ and then do the reverse of the operations that transform $\mathcal{P}^b_{n-1}$ into $\mathrm{CP}_{i,i+1}$. By the following lemma we prove that to obtain $\mathrm{CP}_{i,i+1}$ from $\mathcal{P}^a_{n-1}$ takes $n-3$ flips.

Lemma 4.2.1 *Every path $\mathcal{P}^i_{n-1} \in \mathcal{P}(S)$ which is not canonical can be transformed into a canonical form through flipping of edges in at most $(n-3)$ steps.*

Proof Here v_i is one of the end point of the path $\mathcal{P}^i_{n-1}$. There is a j such that the first $j+2$ vertices of the path are $(v_iv_{i+1}v_{i+2}\cdots v_{i+j}v_{i-1})$ or $(v_iv_{i-1}v_{i-2}\cdots v_{i-j}v_{i+1})$. Without loss of generality consider the former path consisting of vertices $(v_i v_{i+1} v_{i+2} \cdots v_{i+j} v_{i-1} \cdots v_k)$. If $j = n-2$ then we are done because all the edges are on the boundary of the convex hull and hence the path is a canonical path. If $j < n-2$ then we add the edge $(v_i v_{i-1})$ and delete $(v_{i+j} v_{i-1})$. The resulting set of vertices constitutes a path $\mathcal{P}_{n-1}^{i+j}$ because the addition of $(v_i v_{i-1})$ makes a cycle $(v_i v_{i+1} v_{i+2} \cdots v_{i+j} v_{i-1} v_i)$ where the vertex v_{i-1} has degree three. The cycle is broken by removing the edge $v_{i+j} v_{i-1}$ connected to v_{i-1} and reducing its degree to two. So the path $\mathcal{P}_{n-1}^{i+j}$ has the set of $j+2$ vertices $(v_{i+j} v_{i+j-1} \cdots v_i v_{i-1})$. Thus the quality of the path $\mathcal{P}_{n-1}^{i}$ is increased by one.

This implies that if $\mathcal{P}_{n-1}^{q}$ represents any path at step i and $\mathcal{P}_{n-1}^{r}$ is generated from $\mathcal{P}_{n-1}^{q}$ at step $i+1$ then $Qua(\mathcal{P}_{n-1}^{r}) = Qua(\mathcal{P}_{n-1}^{q}) + 1$.

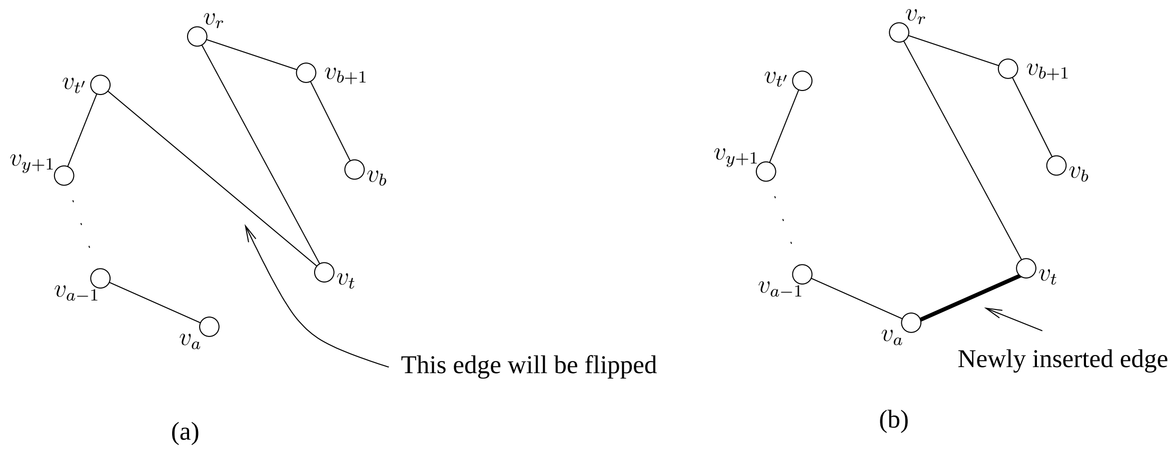


Fig. 4.7: (a) Shows the edge for flip (b) Shows the resultant path after flipping the edge $\{v_t, v_{t'}\}$.

Since there are at least two edges are on boundary for any given path, the number of flips required is $n-3$. ∎

It is shown in the following, that to transform any two paths from one to another the total number of flips is linear.

Theorem 4.2.2 *For any paths $\mathcal{P}^i_{n-1}, \mathcal{P}^j_{n-1} \in \mathcal{P}(S)$, it takes at most $2n-5$ flips to transform one into another.*

Proof From Lemma 5.1.2 it is clear that to obtain a canonical path from any given path $\mathcal{P}^i_{n-1}$ at most $n-3$ flips are necessary. Also, from the definition of canonical path we find that there are n possible canonical paths of n points in convex position since i can take any value from 0 to $n-1$. Moreover, with a single flip it is possible to transform a canonical path into another one. This gives us a total of $n-2$ flips. Similarly, transforming $\mathcal{P}^i_{n-1}$ into canonical form takes $n-3$ flips. Therefore, the total number of flips to transform $\mathcal{P}^i_{n-1}$ into $\mathcal{P}^j_{n-1}$ is at most $2n-5$. ∎

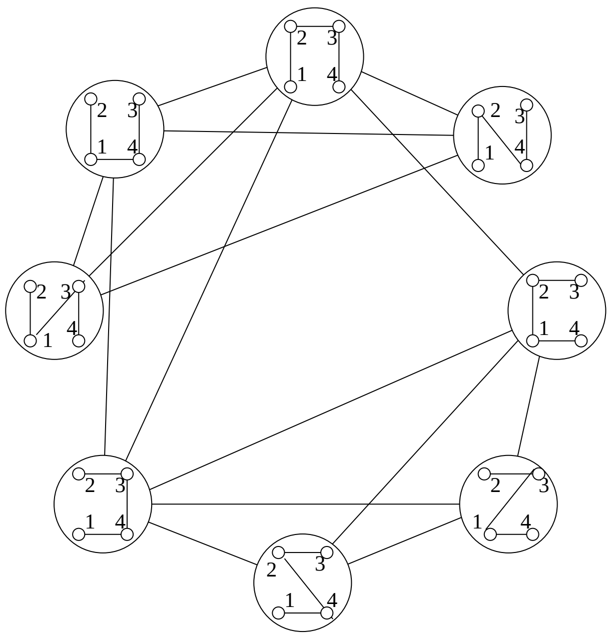
4.2.3 Experimental Results for Path Transformation

In this section, we present experimental results of path transformation considering the set P of n points. Recall that P is a set of n points in general position in the plane.

Theoretically, the proof of connectedness of the path graph whose vertices are non-crossing paths of P, and where two paths $\mathcal{P}^a_{n-1}$ and $\mathcal{P}^b_{n-1}$ are adjacent if $\mathcal{P}^b_{n-1} = \mathcal{P}^a_{n-1} \backslash \{e\} \cup \{f\}$) is difficult. We implemented the problem for a small set of points n ($n \leq 13$) in general position in the plane. The experiment to solve the problem employs a brute-force technique to generate all paths (intersected and intersection free) on n vertices. The number of all such paths is $n!/2$. Any path connecting all vertices of P can be represented by a permutation of $n-$ digits. We filtered out all intersected paths and compute $\mathcal{P}_{accept}$, the set of permutations that correspond to the planar paths of $n-$ vertices.

Denote $P' \in \mathcal{P}_{accept}$ as the initial path consisting of all the vertices with increasing order of their indices. We perform Depth First Search (DFS) technique starting from the initial permutation P' to find out those permutations, each of which differs from P' in such a way that their equivalent path representations differ by one edge. All those permutations (including P') are marked indicating that they can be reached from P' at most one flip. A similar step is followed with all the children of P' where the children of P' are those permutations marked at previous step. This process of marking permutations is continued following Depth First Search strategy until all the permutations are covered (either marked or not marked).

Finally, it is checked whether all the permutations are marked or not. If all of the permutations are marked then the meta graph is connected meaning any path is reachable from any other path. If all the permutations are not marked, then this means the meta graph is not connected. In the experiments we found that in every run with a random set of points, or a specific input-set, the corresponding meta graph is connected. We performed more than 100 simulation runs with point sets of size n ($n \leq 13$). Since the algorithm generates all the paths (planar and non-planar) of n points and accepts only those which are planar, it takes exponential amount of time to generate all such paths. Increasing the number of points (more than 13) costs a lot of time to execute the algorithm. Therefore, we took at most 13 points for our experiments. Each time of the simulation the points were randomly generated in general position with no three points collinear. All the simulation runs produced the same result of connectedness of the corresponding meta graph. We did not find a counter example where the meta graph is not connected. The conclusion that we draw from the results of our experiment lead us to believe that for any value of n the corresponding meta graph is connected.



8 acceptable permutations of 4! permutations

1234
1243
1423
1432
2134
2143
2314
3214

Fig. 4.8: (a) The meta graph of paths with vertices showing all possible paths of 4 points defined in the plane.

As an illustration, Fig. 4.8 shows the meta graph of paths of four vertices placed on the corner of a square (points are shown inside each circle labeled as 1,2,3,4). There are 8 acceptable permutations out of $4!/2 = 12$ permutations as depicted. Two incident vertices with each undirected edge in the graph indicate that the paths they represent are reachable from one another by a single flip. Note that in this particular example one can reach any one vertex from another with at most three flips.

Chapter 5

Counting and Enumeration

5.1 Counting Paths

We show how to count all the directed and undirected paths on a point set in convex position. A simple recurrence formula is used to count all such paths. Recall that $\mathcal{P}_k^i$ denotes the set of paths length k starting at v_i and connecting all vertices of V_k^j for some j where $v_i \in V_k^j$. The number of such paths ($\mathcal{P}_k^i$) is denoted by p_k^i. The basic idea of counting relies on the definition of p_k^i. For any path $P' \in \mathcal{P}_k^i$ (one end is v_i), let v_ℓ be the other end of this path.

Lemma 5.1.1 *For any path $\mathcal{P}' \in \mathcal{P}_k^i$ on V_k^j ($k \leq n-3$) let v_ℓ be the other end of $\mathcal{P}'$. We can construct a new path P'' of length $k+1$, $\mathcal{P}'' \in \mathcal{P}_{k+1}^i$ from $\mathcal{P}'$ by connecting v_ℓ of P' to v_{j-1} or v_{j+k+1}. For $k = n-2$ we can connect v_ℓ only to v_{j+k+1}.*

Proof Consider the end vertex v_ℓ of $\mathcal{P}'$. Assume v_ℓ is connected to any of the vertices other than v_{j-1} or v_{j+k+1} to construct $\mathcal{P}''$. Select $v_m \in V \backslash V_k^j$ such that $v_m \neq v_{j-1}$ and $v_m \neq v_{j+k+1}$. Consider the edge (v_m, v_ℓ). Joining v_m and v_ℓ divides $V \backslash V_k^j$ into two disjoint sets of consecutive vertices $X = \{v_{\ell+1}, v_{\ell+2}, \cdots, v_{m-1}\}$ and $Y = \{v_{m+1}, v_{m+2}, \cdots, v_{j-1}\}$ as shown in Fig. 5.1.

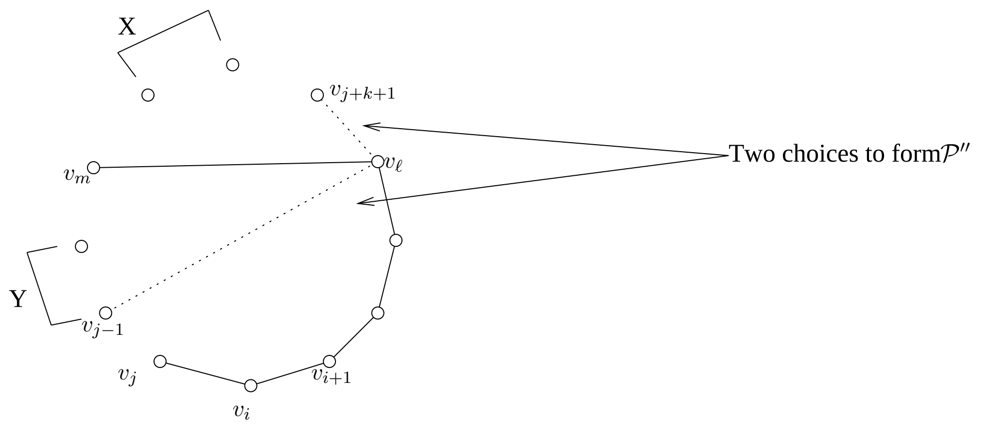


Fig. 5.1: Showing v_ℓ can only be connected to either v_{j-1} or v_{j+k+1} to construct $\mathcal{P}''$ from $\mathcal{P}'$.

Any vertex of either set (X or Y) can see only the vertices of the same set (X or Y) because all the vertices of the other set are blocked by the edge (v_m, v_ℓ). In this circumstance it is obvious that any attempt to connect any vertex $v_r \in X$ to any vertex $v_s \in Y$ will result in a crossing with edge (v_m, v_ℓ). See the Fig. 5.1.

Therefore, the only way to get rid of crossings of edges and have a Hamiltonian crossing-free path $\mathcal{P}''$ of length $k+1$ from $\mathcal{P}'$ is to join the vertex v_ℓ to either of the vertices v_{j-1} or v_{j+k+1} of $V \backslash V_k^j$. If $k = n-2$ then there is only one vertex v_{j+k+1} that can be connected to the other end of $\mathcal{P}'$ to complete $\mathcal{P}''$.

This completes the proof. ▮

Lemma 5.1.2 *The number of directed Hamiltonian paths of length* $n-1$ *and starting at vertex* v_i *is* 2^{n-2}.

Proof We can prove the above lemma through induction.

Recall that p_k^i denotes the number of paths of length k and starting at v_i. For $k=0$, it is trivial that the number of paths of length 0 and starting at v_i equals one as there is only one path beginning at v_i with no edge. For $k=1$, $p_1^i = 2$ because there can be only two paths of length one beginning from v_i namely (v_iv_{i+1}) and (v_iv_{i-1}). Similarly for $k=2$, $p_2^i = 4 = 2p_1^i$ since the total number of paths of length 2 starting from v_i is 4. These paths are $(v_iv_{i+1}v_{i+2})$, $(v_iv_{i+1}v_{i-1})$, $(v_iv_{i-1}v_{i-2})$ and $(v_iv_{i-1}v_{i+1})$. Fig. 5.2 shows all the four paths of length 2 and starting from v_i.

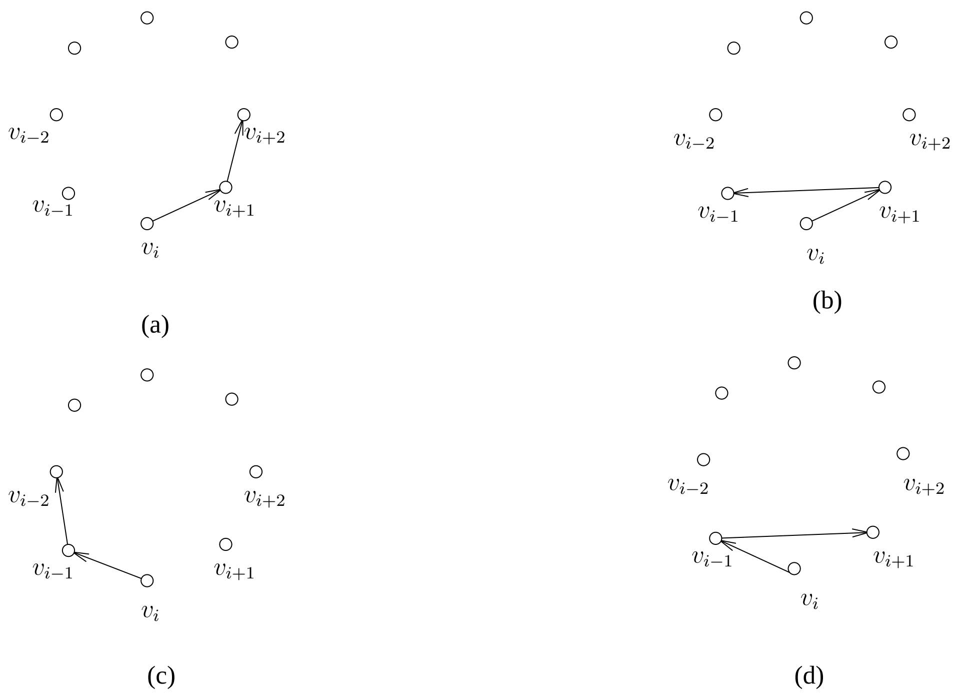


Fig. 5.2: All paths of length 2 starting at v_i are shown in (a) $(v_iv_{i+1}v_{i+2})$ (b) $(v_iv_{i+1}v_{i-1})$ (c) $(v_iv_{i-1}v_{i-2})$ and (d) $(v_iv_{i-1}v_{i+1})$

For every value of k ($k \in \{0, 1, \cdots, n-3, n-2\}$) there are twice as many paths for p_k^i than p_{k-1}^i. This is because for each of the path p_{k-1}^i of length $k-1$ we can form two more paths of length k with the other end of each of the path p_{k-1}^i. So we have $p_k^i = 2p_{k-1}^i$ for $0 < k \leq n-2$.

For the value $k = n-1$ we have $p_{n-1}^i = p_{n-2}^i$, because for each path of length $n-2$ we get exactly one corresponding path of length $n-1$. Evaluating the recursion for $k = n-1$ the number of directed paths equals to $p_{n-1}^i = 2^{n-2}$. ▮

Lemma 5.1.3 *All the paths of $\mathcal{P}_{k+1}^i$ derived from $\mathcal{P}_k^i$ are distinct for $0 \leq k \leq n-2$.*

Proof Take any path $\mathcal{P} \in \mathcal{P}_k^i$ that starts at v_i and uses all the vertices of $V_k^j = \{v_j, v_{j+1}, \cdots, v_{j+k}\}$ where $v_i \in V_k^j$. The path P can produce two paths of length $k+1$, say P' and P''. Assume P' uses the vertex set, $V_1 = \{v_j, v_{j+1}, \cdots, v_i, v_{i+1}, \cdots, v_{j+k}, v_{j+k+1}\}$ and P'' uses $V_2 = \{v_{j-1}, v_j, \cdots, v_i, v_{i+1}, \cdots, v_{j+k}, v_{j+k}\}$. Since $V_1 \neq V_2$, P' and P'' are different.

From above we can conclude that although P' and P'' use different sets of vertices, their starting edges, i.e., v_iv_{i+1} is common to them as they are generated from P in which the starting edge v_iv_{i+1} is not changed to form P' and P''. Now consider another path $\mathcal{Q} \in \mathcal{P}_k^i$ which produces two paths Q' and Q'' of length $k+1$. Since the starting edge of Q is not equivalent to that of P, all the paths Q' and Q'' of Q are different from P' and P'' of P. ▮

Now we can prove the main theorem:

Theorem 5.1.4 *The number of undirected Hamiltonian paths of length $n-1$ is $n2^{n-3}$.*

From Lemma 5.1.2 it has been found that the number of directed paths (of length $n-1$) of S starting at vertex v_j is equal to 2^{n-2}. Since there are n vertices, applying the procedure described in the above lemma for each of the individual vertices the total count of directed paths equals $n2^{n-2}$. On the other hand for the case of counting the number of undirected paths, observe that each of the paths is counted twice, one starting with the higher labeled vertex and the same path with the lower labeled vertex. So, the number of undirected paths is $n2^{n-3}$. ▮

5.2 Enumeration of Paths

In this section and the section following that we present two algorithms to generate all the paths of $n-$ points when the point set is convex position. Both algorithms offer their relative advantages and limitations. The first algorithm (described in this section) is recursive. In some sense, this algorithm is simple and straightforward, uses only single flips to produce all the paths of n points. But the main limitation is that it takes an exponential amount of space because of its recursive nature (uses stack to store intermediate paths). In order to overcome the limitation of this algorithm we present a non-recursive algorithm. This non-recursive algorithm uses less space. It takes linear space to store the paths and produce new paths from them. The non-recursive algorithm employs flips of size one and two. As defined earlier, a flip of size two means removing two edges and putting two edges at a time in a path. Following the presentations of the algorithms then space and time complexities are derived.

In the recursive algorithm, our strategy to generate all the paths of $\mathcal{P}(S)$ is to begin with a canonical path, $\mathrm{CP}_{i,i+1} \in \mathcal{P}(S)$ and generate all those paths possible from $\mathrm{CP}_{i,i+1}$ with edge (v_i, v_{i+1}) fixed. The notion of vertex visibility plays the vital role to carry out the process of path generation. As we go through a recursive procedure to obtain new paths, consider building a tree where the root vertex of the tree represents the path $\mathrm{CP}_{i,i+1}$ and all the children of $\mathrm{CP}_{i,i+1}$ are designated as the vertices of the tree.

The spawning of a new path can only be initiated from its parent when a vertex v_i in the parent path can see at least one vertex v_j such that $j > i + 1$.

Lemma 5.2.1 *There is an algorithm that produces exactly* 2^{n-3} *distinct Hamiltonian paths starting from a canonical path.*

Proof Begin with a canonical path $\mathrm{CP}_{i,i+1}$ and consider it the root of the tree. Consider the vertex v_{i-1} of $\mathrm{CP}_{i,i+1}$.

The vertices visible to v_{i-1} are $v_i, v_{i+1}, v_{i+2}, v_{i+3}, \cdots, v_{i-3}$ and the number of those vertices is $n-2$ since v_{i-1} can not see itself and v_{i-2}. Among the set of visible vertices consider the second furthest visible vertex v_{i+1}.

Start by generating the first path from $\mathrm{CP}_{i,i+1}$ by flipping the edge between the second furthest visible vertex v_{i-2} and v_{i-3}, i.e. removing edge (v_{i+1}, v_{i+2}) and inserting an edge (v_{i-1}, v_{i+1}). Represent this child path as $\mathcal{P}_{n-3}(v_{i+2}) = \mathrm{CP}_{i,i+1} \cup (v_{i-1}, v_{i+1}) \backslash (v_{i+1}, v_{i+2})$ that differs from its parent by a single edge. Here $\mathcal{P}_j(v_i)$ denotes a path where j is the number vertices visible to v_i. Call the edge (v_{i-1}, v_{i+1}) as a *mark* of $\mathcal{P}_{n-3}(v_{i+2})$. A mark is an edge that is not flipped, i.e., it remains unchanged for a parent and all its successors.

Place this newly generated path as the leftmost child of the root. Continue generating the rest of the $n-3$ paths and placing them from left to right:

$$\begin{aligned}
\mathcal{P}_{n-3}(v_{i+2}) &= \mathrm{CP}_{i,i+1} \cup (v_{i-1}, v_{i+1}) \backslash (v_{i+1}, v_{i+2}) \\
\mathcal{P}_{n-4}(v_{i+3}) &= \mathrm{CP}_{i,i+1} \cup (v_{i-1}, v_{i+2}) \backslash (v_{i+2}, v_{i+3}) \\
\mathcal{P}_{n-5}(v_{i+4}) &= \mathrm{CP}_{i,i+1} \cup (v_{i-1}, v_{i+3}) \backslash (v_{i+3}, v_{i+4}) \\
&\vdots \\
\mathcal{P}_{2}(v_{i-3}) &= \mathrm{CP}_{i,i+1} \cup (v_{i-1}, v_{i-4}) \backslash (v_{i-4}, v_{i-3}) \\
\mathcal{P}_{1}(v_{i-2}) &= \mathrm{CP}_{i,i+1} \cup (v_{i-1}, v_{i-3}) \backslash (v_{i-3}, v_{i-2}).
\end{aligned}$$

Note that the marks of paths $\mathcal{P}_{n-3}(v_{i+2}), \mathcal{P}_{n-4}(v_{i+3}), \cdots, \mathcal{P}_1(v_{i-2})$ are (v_{i-1}, v_{i+1}), $(v_{i-1}, v_{i+2}), \cdots, (v_{i-1}, v_{i-3})$, respectively. Since each of the marks is unique to each path at this level (level-1, root is considered at level-0), all the paths generated are unique, i.e. no path at this level has the same marks.

Starting from the leftmost child of the root we observe that the end vertex v_{i+2} of the path $\mathcal{P}_{n-3}(v_{i+2})$ can see $n-3$ vertices (which is one less than the number that its parent can see). This is because v_i is now blocked by edge $v_{i-1}v_{i+2}$. Like the previous step, the leftmost child representing the path with end vertex v_{i-3} can produce $n-4$ paths as its children in the next level where each of the paths (already having a mark edge (v_{i-1}, v_{i+1})) is generated with an additional mark edge (v_{i-1}, v_{i+2}). These two marks uniquely identify a path which was not generated at a previous step and also distinct in this level (level-2).

Similarly, the process is continued with each of the children of the root up to the rightmost child which can see only v_{i-3} and can not generate any path.

Since at each level a new mark is added to each of the paths (which consists of a set of unique marks from all previous levels) it can be concluded that all the paths in the tree are uniquely generated. This recursive construction of paths continues until the leftmost child of the tree can see only one vertex and thus can not produce any offspring. The scenario is depicted in Fig. 5.3.

Let $N_j(v_i)$ be the number of child paths produced by $\mathcal{P}_j(v_i)$.

The following gives us the values of $N_j(v_i)$s:

$$\begin{aligned} N_1(v_{i-2}) &= 0 \\ N_2(v_{i-3}) &= 1. \end{aligned}$$

Continue going from the third rightmost vertex and so on and apply recursion in each step:

$$\begin{aligned} N_3(v_{i-4}) &= N_2(v_{i-3}) + N_1(v_{i-2}) \\ &= (1+1) + (1+0) \\ &= (1+2^1-1) + (1+2^0-1) \end{aligned}$$

The first 1 in each of the terms of the sum indicates the presence of the vertices

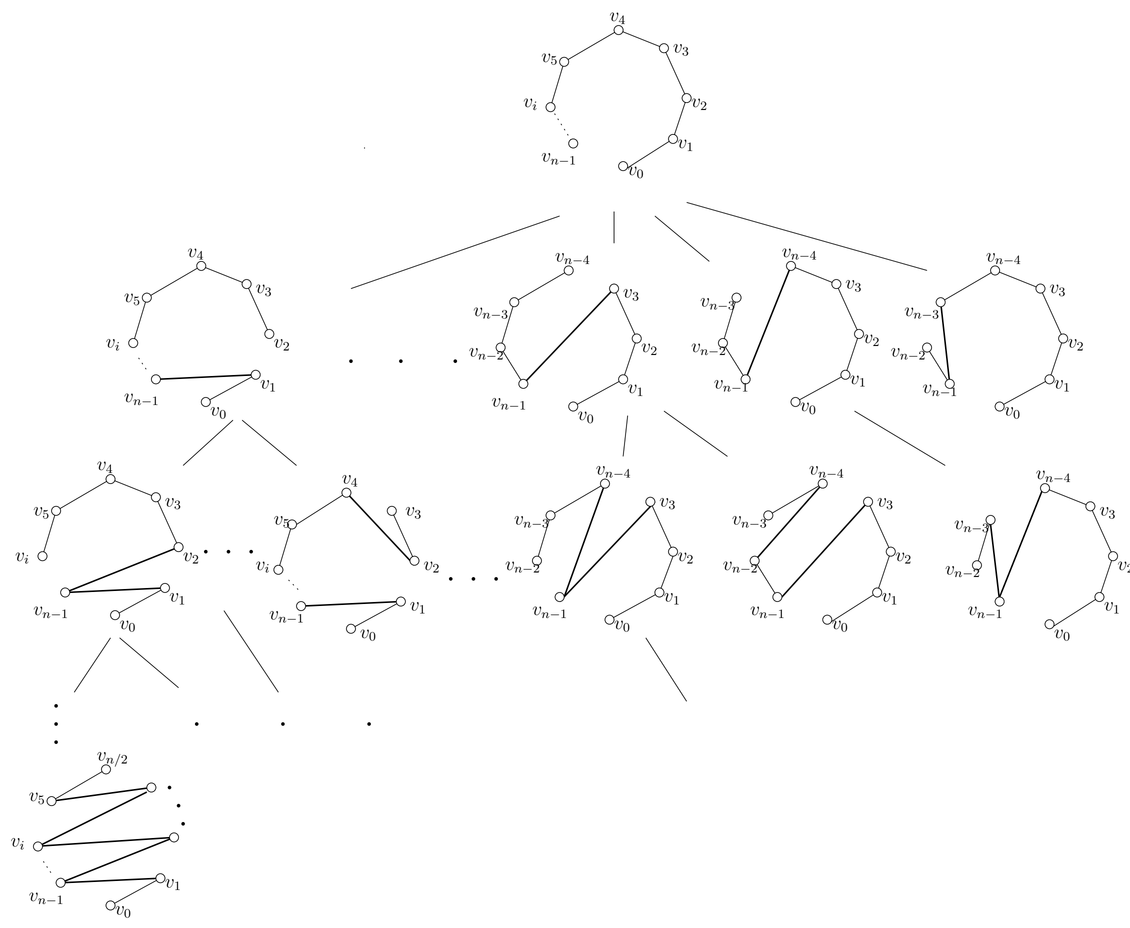


Fig. 5.3: Recursive enumeration of all paths from a canonical path $\mathrm{CP}_{0,n-1}$. Recursion terminates as vertex $v_{n/2}$ of the leftmost leaf of the tree can see only one vertex and thus is incapable of producing any path.

themselves and the rest of the integers denote the number of children they produce.

$$\begin{aligned}
N_4(v_{i-5}) &= N_3(v_{i-4}) + N_2(v_{i-3}) + N_1(v_{i-2}) \\
&= (1+3) + (1+1) + (1+0) \\
&= (1+2^2-1) + (1+2^1-1) + (1+2^0-1) \\
&= (2^2) + (2^1) + (2^0) \\
&\vdots
\end{aligned}$$

$$
\begin{aligned}
N_{n-2}(v_i) &= N_{n-3}(v_{i+2}) + N_{n-4}(v_{i+3}) + N_{n-5}(v_{i=4}) + \cdots + N_2(v_{i-3}) + N_1(v_{i-2}) \\
&= 2^{n-4} + 2^{n-5} + 2^{n-6} + \cdots + 2^1 + 2^0 \\
&= 2^{n-3} - 1
\end{aligned}
$$

This is the number of total vertices (excluding the root) of the tree. Counting the root together with this number we get 2^{n-3} which is the total number of paths that can be generated by $\text{CP}_{i,i+1}$. ▮

Consider the example of a set of $n = 6$ vertices given in Fig. 5.4. The root of the tree is the path $\text{CP}_{1,2}$ ($v_1v_2v_3v_4v_5v_0$). The generation of all paths from this canonical path ($2^{6-3} = 8$) starting with edge v_1v_2 is depicted.

5.2.1 Recursive Generation of All Paths

In the last section (specifically Lemma 5.2.1) we showed how we could generate paths from a single canonical path. Here we describe the algorithm of generating all paths of n vertices in convex position. We use all canonical paths $\text{CP}_{i,i+1}$ and $\text{CP}_{i+1,i}$ (there are $2n$ such canonical paths) to enumerate all paths. As our algorithm uniquely generates paths, we need to keep track, as the algorithm progresses whether a particular path is generated twice. For this we form a $A = n$ x n matrix that updates its entry $a_{i,j} = 1$ where $j = i + 1$ or $j = i - 1$ if all the paths beginning with edge (v_i, v_j) are generated else $a_{i,j} = 0$.

Start first with $\text{CP}_{0,1}$ to generate all paths from it and then with $\text{CP}_{1,0}$, $\text{CP}_{1,2}$

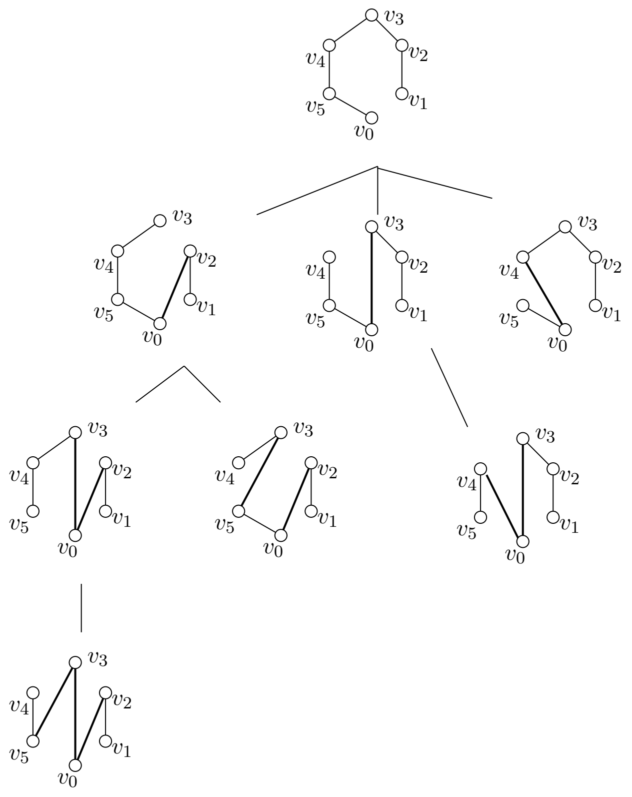


Fig. 5.4: Recursive enumeration of all paths from the canonical path $\mathrm{CP}_{1,2}$ ($v_1v_2v_3v_4v_5v_0$) of 6 vertices. Bold edges represent marks of the corresponding paths.

$\mathrm{CP}_{2,1}$, and continue up to $\mathrm{CP}_{0,n-1}$ and $\mathrm{CP}_{n-1,0}$. In general for each i (starting i at 0 and then incrementing it by one after each pair $\mathrm{CP}_{i,i+1}$ and $\mathrm{CP}_{i+1,i}$ are done with generation until $i = n-1$) we generate paths from $\mathrm{CP}_{i,i+1}$ and $\mathrm{CP}_{i+1,i}$ and continue until paths are generated from $\mathrm{CP}_{0,n-1}$ and $\mathrm{CP}_{n-1,0}$.

Beginning with $\mathrm{CP}_{0,1}$ enumerate all the paths recursively as described in Lemma 5.2.1. And this generates 2^{n-3} paths each of which begins with edge (v_0, v_1). The entry $a_{0,1}$ in the matrix A is updated to 1 as all the paths from $\mathrm{CP}_{0,1}$ are generated. Then begin and continue with $\mathrm{CP}_{1,0}, \mathrm{CP}_{1,2}, \cdots, \mathrm{CP}_{0,n-1}$ and follow the same procedure.

Resolve the difficulty of avoiding previously generated paths in the following way.

Suppose we are generating paths from $\mathrm{CP}_{i,i+1}$. Let $\mathcal{P}'$ be a path generated from $\mathrm{CP}_{i,i+1}$. Let (v_p, v_q) be its end-edge ($q = p + 1$ or $q = p - 1$). If the entry $a_{p,q}$ of A is 0 then none of the paths with end-edge (v_p, v_q) have been generated then P' is not duplicated and hence can be outputted. If on the other hand, the entry $a_{p,q}$ is 1, then we do not ouput P'.

5.2.2 Space and time complexity

As described earlier, we begin with a canonical path and start generating all the paths from it in the next level. And then do the same of generating paths from each of the paths (generated so far) in the next level and so on until we are finished with the last one from which no paths can be generated. We need to keep track all the paths generated from the root up to the left-most leaf to continue generating paths which produce exponential number of paths (2^{n-3} paths). It follows that the space complexity is also exponential which is $O(2^n)$.

In the case of time complexity we find the maximum time required to successively generate any two paths P_i and P_{i+1}. If $t(P_i)$ represents the time elapsed from the beginning of the algorithm until P_i generated then $max_i = t(P_{i+1}) - t(P_i)$ determines the amount of time required to generate and output two successive paths P_i and P_{i+1}. Since there can be an exponential number of paths that are generated (according to the algorithm) but not outputted between P_i and P_{i+1}, the time max_i results in exponential. This gives $O(2^n)$ time complexity.

5.3 Non-recursive Construction

Here we discuss an algorithm to non-recursively generating all paths $\mathcal{P}(S)$.

Each path consisting of n vertices is unique and it can be represented by a permutation of $n-$ digits where each of the vertices represents a digit.

Begin with the canonical path $\text{CP}_{0,1}$ $(v_0v_1v_2v_3\cdots v_{n-2}v_{n-1})$. What we wish to do is enumerate all 2^{n-3} paths from $\text{CP}_{0,1}$ keeping the edge (v_0, v_1) fixed (not flipped) and flipping edges (v_{i-1}, v_i) not involving (v_0, v_1). We will also do the same with the rest of the canonical paths $\text{CP}_{m,m+1}$ and $\text{CP}_{m+1,m}$ by not flipping the edges (v_m, v_{m+1}) and (v_{m+1}, v_m) respectively. Assume the equivalent $n-$digit representation of the permutation of $\text{CP}_{0,1}$ $(v_0v_1v_2v_3\cdots v_{n-2}v_{n-1})$ is $A_{0,1} = a_0a_1a_2\cdots a_{n-2}a_{n-1}$ where $a_{n-1} > a_{n-2} > a_{n-3}\cdots > a_1 > a_0$. Call this the initial permutation where each digit a_i where $0 \leq i \leq n-1$ of the permutation corresponds to each vertex v_i. In the following the generation of successive permutations is shown. Each of the permutations represents a unique path.

5.3.1 Generation Method

Keep the positions of a_0 and a_1 unchanged and change the positions of the remaining digits a_i, $1 < i \leq n-1$, to generate successive permutations.

The basic idea is that, at any step, a digit a_i of the permutation can only have digit $max\{a_{i+1}, a_{i+2}, \cdots, a_{n-1}\}$ or $min\{a_{i+1}, a_{i+2}, \cdots, a_{n-1}\}$ as its right neighbour. Having a minimum or a maximum digit to the right of a digit ensures planarity of the path represented by the permutation (this can be verified from Lemma 5.1.1). For example, $\underline{01}$ 276345 is a permutation of this kind where the digit 7 has 6 as its right neighbour which is the maxmimum of $\{6,3,4,5\}$ and 6 has 3 as its right neighbour which is the minimum of $\{3,4,5\}$ etc.

Find i from the rightmost position of the permutation such that $a_{i-1} < a_i$. If $i = n-1$, obtain the next permutation by exchanging the positions of a_{i-1} and a_i which is equivalent to a single flip (a flip of size one). Let the number of digits to the right of a_i be k. If $i \neq n-1$, then the next permutation is obtained by first exchanging the positions of a_{i-1} and a_i and then reversing the positions of $a_{i+1}, a_{i+2}, a_{i+3}, \cdots, a_{n-1}$. The changing of positions of the digits is equivalent to a flip of size two. The edges that are removed are $(v_{a_i}, v_{a_{i+1}})$ and $(v_{a_{i-1}}, v_{a_{i-2}})$ and the newly inserted edges are $(v_{a_{i-1}}, v_{a_1})$ and $(v_{a_{i-2}}, v_{a_i})$ which is the reflection of the above permutation. The process of generating a permutation P_m from a previous one P_{m-1} beginning from the rightmost digit of P_{m-1} to find $a_{i-1} < a_i$ is continued until we obtain $a_2 > a_3 > a_4 \cdots > a_{n-2} > a_{n-1}$.

As an example, consider $n = 8$. All the possible permutations of $8-$digits (from Lemma 5.1, $2^{8-3} = 32$) according to this method are shown below (keeping the positions of the two leftmost ($\underline{01}$) digits unchanged):

$\underline{01}$ 234567 $\rightarrow$ $\underline{01}$ 234576 $\rightarrow$ $\underline{01}$ 234756 $\rightarrow$ $\underline{01}$ 234765 $\rightarrow$ $\underline{01}$ 237456 $\rightarrow$ $\underline{01}$ 237465

$\underline{01}$ 237645 $\rightarrow$ $\underline{01}$ 237654 $\rightarrow$ $\underline{01}$ 273456 $\rightarrow$ $\underline{01}$ 273465 $\rightarrow$ $\underline{01}$ 273645 $\rightarrow$ $\underline{01}$ 273654

$\underline{01}\ 276345 \rightarrow \underline{01}\ 276354 \rightarrow \underline{01}\ 276534 \rightarrow \underline{01}\ 276543 \rightarrow \underline{01}\ 723456 \rightarrow \underline{01}\ 723465$

$\underline{01}\ 723645 \rightarrow \underline{01}\ 723654 \rightarrow \underline{01}\ 726345 \rightarrow \underline{01}\ 726354 \rightarrow \underline{01}\ 726534 \rightarrow \underline{01}\ 726543$

$\underline{01}\ 762345 \rightarrow \underline{01}\ 762354 \rightarrow \underline{01}\ 762534 \rightarrow \underline{01}\ 762543 \rightarrow \underline{01}\ 765234 \rightarrow \underline{01}\ 765243$

$\underline{01}\ 765423 \rightarrow \underline{01}\ 765432$.

5.3.2 Correctness of the Algorithm

What we wish to show is that all the paths generated by this algorithm are unique and the total number of such paths is 2^{n-3}. We can imagine a binary tree where each of the vertices will represent a digit a_i of a permutation. Let us assume $\mathcal{A} = \{a_1, a_2, \cdots, a_{n-1}\}$ and let the root be denoted as the element a_1 which is at level-0. At the next level the left and right children of the root are determined by $max\{\mathcal{A} - \{a_1\}\}$ and $min\{\mathcal{A} - \{a_1\}\}$, respectively.

In general at level-i the left and right children of a vertex at level-$(i-1)$ will be determined in the following way (see Fig. 5.5 where $a_0a_1a_2a_3a_4a_5a_6 = 0123456$):

left child $= max\{\mathcal{A} - \{$*all* $a_i s$ *along the path from the root to the current vertex at level i-1*$\}\}$

and

right child $= min\{\mathcal{A} - \{$*all* $a_i s$ *along the path from the root to the current vertex at level i-1*$\}$

As there are $n-1$ elements in the permutation $a_1a_2a_3 \cdots a_{n-2}a_{n-1}$ and the parent of each leaf contains exactly one leaf, the height of the tree is $n-1-2 = n-3$. Then

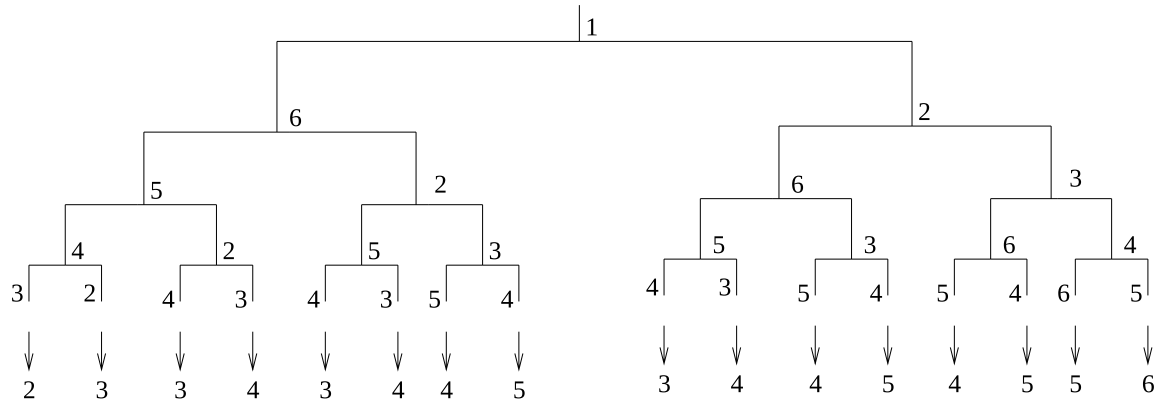


Fig. 5.5: Tree representation of permutations

the total number of leaves equals 2^{n-3}. As all the elements of $\mathcal{A}$ are distinct and each path from a leaf to the root is unique, all such paths of the tree are unique.

5.3.3 Mapping Between $\mathbf{CP}_{0,1}$ and $\mathbf{CP}_{m,m+1}$ ($\mathbf{CP}_{m+1,m}$)

After generating all the valid permutations (each represents a unique path) of $A_{0,1}$ we can now proceed to generate other paths from each of the remaining canonical paths. Without loss of generality we only show mapping between $\mathrm{CP}_{0,1}$ and $\mathrm{CP}_{m,m+1}$. Mapping between $\mathrm{CP}_{0,1}$ and $\mathrm{CP}_{m+1,m}$ is also meant subsequently after mapping of $\mathrm{CP}_{m,m+1}$ is done, although not mentioned.

There is a one-to-one correspondence between $\mathrm{CP}_{0,1}$ and all other canonical paths $\mathrm{CP}_{m,m+1}$ and $\mathrm{CP}_{m+1,m}$. Let the equivalent $n-$ digit representation of the permutation of $\mathrm{CP}_{m,m+1}$ be $A'_{m,m+1} = a_m a_{m+1} a_{m+2} \cdots a_{m-2} a_{m-1}$.

In order to enumerate paths from $\mathrm{CP}_{m,m+1}$ we first map $A'_{m,m+1}$ (the equivalent permutation of $\mathrm{CP}_{m,m+1}$) and $A_{0,1}$ (the equivalent permutation of $\mathrm{CP}_{0,1}$) with the function $f : A'_{m,m+1} \rightarrow A_{0,1}$ such that the function yields:

$$f(a_{i_m}) = a_{i_0}, \quad f(a_{i_{m+1}}) = a_{i_1}, \quad \cdots \quad , f(a_{i_{m-1}}) = a_{i_{n-1}}.$$

A mapping between $A'_{4,3}$ and $A_{0,1}$ is shown in Fig. 5.6.

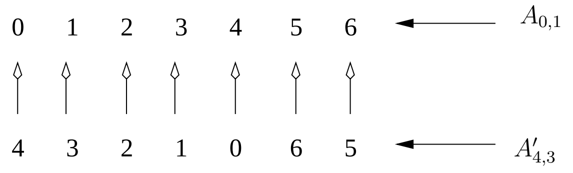


Fig. 5.6: Mapping between $A'_{4,3}$ and $A_{0,1}$.

Start generating permutations with $A_{0,1}$ as it is done before. But this time we need to check whether paths represented by permutations are already generated or not. In generating permutations, if we find any of them is already generated then the permutation is discarded, else it is generated. To realize the fact whether any paths are already generated we introduce the term what we call *index* to the two leftmost (rightmost) digits of all the permutations. Index starts from the two leftmost digits of $A_{0,1}$ with the value 1. That is, the index of the two leftmost digits a_0a_1 of a permutation is 1. Next the index of the two digits a_1a_0 is 2, the index of a_1a_2 is 3 and so on. So in general for each m (starting at $m = 0$ and then incrementing it by one after each pair of digits a_ma_{m+1} and $a_{m+1}a_m$ are indexed until $m = n - 1$) the indices of two leftmost (rightmost) digits a_ma_{m+1} and $a_{m+1}a_m$ are $2m + 1$ and $2m + 2$ respectively. For example, if $n = 6$ the indices of two leftmost (rightmost) digits $a_0a_1, a_1a_0, a_1a_2, a_2a_1, a_2a_3, a_3a_2, a_3a_4, a_4a_3, a_4a_5, a_5a_4, a_5a_0, a_0a_5$ are

$1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12$, respectively.

As a permutation is generated from $A_{0,1}$, do the inverse of f on this permutation and see whether the index of the two rightmost digits or their reverse is smaller than the index of the two leftmost digits of this permutation. If this is smaller then this means that the permutation is already generated and we can discard the permutation. This is evident since we begin generating all the permutations in order where the two leftmost digits of all the permutations are $a_0a_1, a_1a_0, a_1a_2, a_2a_1, \cdots, a_{n-1}a_{n-2}, a_0a_{n-1}$. This ensures us that when we find the index of the two rightmost (or their reverse) digits is smaller than the index of two the leftmost digits, this indicates that those permutations with smaller index in the two leftmost digits are already generated.

Consider the example with $n = 6$ and assume that we are going to generate paths from CP$_{3,4}$ where $A'_{3,4} = \underline{34}\ 5012$ and $A_{0,1} = \underline{01}\ 2345$. First we do $f : A'_{3,4} \rightarrow A_{0,1}$ and find $f(3) = 0$, $f(4) = 1$, $f(5) = 2$, $f(0) = 3$, $f(1) = 4$, $f(2) = 5$. So the first permutation is $\underline{01}\ 2345$. As this is obtained we need to check whether it is already generated. Applying the reverse function we get $f^{-1}(0) = 3$, $f^{-1}(1) = 4$, $f^{-1}(2) = 5$, $f^{-1}(3) = 0$, $f^{-1}(4) = 1$, $f^{-1}(5) = 2$, i.e., the permutation is $\underline{34}\ 5012$ where the index of the two rightmost digits is smaller than the index of the two leftmost digits (index of a_1a_2 is 3 and the index of a_3a_4 is 7). This means that we will produce this permutation as output since it is generated with $A'_{1,2}$. Let us assume another permutation $\underline{01}\ 5423$. Now apply the reverse function and get the following permutation $\underline{34}\ 2150$. Since (index of a_5a_0 is 11 and the index of a_3a_4 is 7) we are sure that this permutation has not been generated before and we can generate this permutation.

Note that we do not need to generate any permutations from $A'_{0,n-1}$ since all the permutations that can be generated from $A'_{0,n-1}$ have their two rightmost digits (or their reverse) equal to the two leftmost digits of all the permutations that have already been generated. In fact, permutations from $A'_{n-2,n-1}$ and $A'_{n-1,n-2}$ do not need to be generated for the same reasoning.

5.3.4 Space and time complexity

Space complexity has been improved a lot (from exponential to linear) in this non-recursive procedure. We begin with a canonical path and start generating a path from it. And then generate a new path from the currently generated one in sequence and so on until there is no more path generated. In the procedure, we do not keep track any of the previously generated path only the current one. Since a path of length $n-1$ of takes $n-$ points to be stored, the space complexity is $O(n)$.

In order to generate and output a permutation P_{i+1} that represents a planar path from a permutation P_i, we generate a permutation P_i and check the two indices of the two leftmost and rightmost digits of the generated permutation. If they satisfy the condition (if the index of the two rightmost digits is greater than the index of the two leftmost digits) then we output the permutation as P_{i+1} otherwise this permutation is not outputted. In the worst case, there can be an exponential number of permutations between P_i and P_{i+1} that are generated but not outputted. This indicates that the time required to produce P_{i+1} when P_i is already found is exponential. Hence the time complexity is factorial.

5.3.5 Implementation of the algorithm

In order to ensure that the proposed non-recursive algorithm works properly and enumerates all the Hamiltonian paths with points in convex position, we implemented the algorithm. The input to the algorithm consists only the number of points (the number of points taken was less than or equal to 12). The algorithm enumerates all the paths by producing the corresponding permutations of the paths without duplication. We performed more than 50 runs and verified the total number of paths and their uniqueness. The algorithm worked correctly.

Chapter 6

Conclusion

In this thesis, attention was directed towards transforming two geometric figures namely, planar paths and planar trees, for certain sets of point through the use of edge replacements commonly called *flip*s. As defined earlier, a flip can be considered as a basic unit of operation for geometric transformation which involves removal of one edge and insertion of a new edge into the given figure such that the resultant figure belongs to the same class as the original one.

We attempted to answer the question whether it is possible to transform a given tree (or path) to another tree (or path) for a fixed set of points in the plane. In the case of tree transformation the answer was given in the affirmative by providing certain theoretical proofs. In path transformation we present experimental results for the connectedness of the meta graph of paths for $n \leq 13$ points. We performed more than 100 simulations with random sets of points and we did not find any counter example for which the meta graph of paths is not connected.

In chapter 4 a method was presented for tree transformation through flips when the points are in general position. In this method, a definition of a canonical tree was given for a fixed set of points in the plane. With the use of the canonical tree we obtained the necessary transformation from one tree to another. Later, description and formation of the meta graph $\mathcal{G}_{\mathcal{T}}$ consisting of vertices which are the non-crossing set of trees $\mathcal{T}(P)$ of a point set P was provided. This helped derive the bound of the diameter of the meta graph in terms of the number flips, which is the shortest longest distance between any two vertices of the meta graph. The diameter of $\mathcal{G}_{\mathcal{T}}$ was also determined to be bounded by $2n - k - s - 2$, where $k, s \geq 1$.

In a similar fashion, the problem of crossing-free Hamiltonian path transformation for points in general position and points in convex position was considered. In the case where the points are in convex position it is proved that at most $(2n-5)$ edge changes are sufficient for any two non-crossing paths on convex postion to be transformed one to another. The main implication of this result is that the meta graph $\mathcal{G}_{\mathcal{P}}$ (defined similarly to $\mathcal{G}_{\mathcal{T}}$) is connected and the diameter bounded by $2n - 5$.

Later in chapter 4, the experimental results of the above problem (path transformation) with a small number of points ($n \leq 13$) in general position were presented. We found that for all the simulation runs (for each run points were chosen randomly in general position in the Euclidean plane) the corresponding meta graphs were connected. Although we can not claim strongly that all the meta graphs are connected as the experiments did not include all the possible point sets, the results gave us the flavour that the connectedness of the meta graphs might be promising. It remains an open problem to show whether the meta graph, $\mathcal{G}_{\mathcal{P}}$ is connected for the set of points in general position.

In chapter 5 we investigated the possibility whether flipping can be used as a technique to enumerate all the geometric objects of a certain set. As an example we made an attempt to count and generate all labelled paths considering the points in convex position.

As, in general, counting precedes generation, we first provided a simple recursion procedure to count the total number of paths and proved that this number equals $n2^{n-3}$. Two algorithms for generating all such paths are explained with example. The first one is recursive, where it was shown that flipping of edges can be used to generate all paths without duplicates. Secondly, we explained a non-recursive algorithm that establishes a one-to-one relation between n vertices and permutations of n digits and allows us to produce uniquely the set of all paths each of which resembles a permutation of n digits. In this non-recursive construction, along with flip of size one, we took advantage of flips of size two to generate all the paths uniquely.

.

Bibliography

[1] K. Appel and W. Haken, Every planar map is four colorable, Bull. American Math. Soc. 82:711-712, 1976.

[2] N. Robertson, D.P. Sanders, P.D. Seymour and R.Thomas, A new proof of the four colour theorem, Electron. Res. Announcement. Amer. Math. Soc. 2:17-25,1996.

[3] D. Avis, K. Fukuda, Reverse search for enumeration, Discrete Applied Mathematics 65(1996)-21-46.

[4] D. Avis, K. Fukuda, A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra, Discrete Computer Geometry 8(3):295-313, 1992.

[5] K. Fukuda and V. Rosta, Combinatorial face enumeration in convex polytopes, Computational Geometry Theory Applied 4:191-198,1994.

[6] K. Fukuda, S. Saito and A. Tamura, Combinatorial face enumeration in arrangements and oriented matroids, Discrete Applied Mathematics 31(2):141-149, 1991.

[7] C.L. Lawson, Transforming triangulations, Discrete Mathematics 3:365-372, 1972.

[8] S. Fortune, A note on Delaunay diagonal flips, AT&T bell Laboratories, Murray Hill, NJ (1987).

[9] A.V. Aho, J.E. Hopcroft and J.D. Ullman, data Structures and Algorithms (Addison-Wesley, Reading, AM, 1987).

[10] H. Telly, Static and dynamic weighted Delaunay Triangulation in the Euclidean Plane and in the flat torus, Research Report UIUCDCS-R-90-1662, Department of Computer Science, University of Illinois at Urbana-Champaign(1990).

[11] B.D. McKay, Isomorph-free exhaustive generation, Journal of Algorithms 26(1998) 306-324.

[12] S. Nakano Efficient generation of triconnected plane triangulations, Computational geometry 27(2004) 109-122.

[13] H. Edelsbrunner, Algorithms in Combinatorial Geometry (springer, Berlin, 1987).

[14] D. Avis, Generating rooted triangulations without repetitions, Algorithmica 16(1996) 618-632.

[15] H. Cormen, et al, Introduction to Algorithms, The MIT Press and McGraw-Hill, 1990.

[16] C. Hernando, M.E. Houle and F. Hurtado, On local transformation of polygons with visibility properties, Theoretical Computer Science 289(2):919-937, 2002.

[17] J.D. Boissonnat, Geometric Structures for Three Dimensional Shape Representation, ACM Transactions on Graphics 3 (1984) 266-286.

[18] J.C. Cavendish, Automatic Triangulation of Arbitrary Planar Domains for Finite Element Method, International Journal for Numerical Methods in Engineering 8 (1974) 679-696.

[19] H.J. Choi, An implementation of a Flight Path Visualization System using Optimization Algorithms of the Triangulated Irregular Network, Ms. Thesis, POSTECH, 1996.

[20] R.C. Read, R.E. Tarjan, Bounds on backtrack algorithms for listing cycles, paths and spanning trees, networks 5(1975) 237-252.

[21] C.A. Holzmann, F. Harary, On the tree graph matroid, SIAM Journal of Applied Mathematics 22(1972), 187-193.

[22] G. Liu, On connectivities of tree graphs, Journal of Graph Theory 12(1988) 453-459.

[23] W. Goddard, H.C. Swart, Distances between graphs under edge operations, Discrete Mathematics 161(1996), 121-132.

[24] R.L. Cummings, Hamilton circuits in tree graphs, IEEE Transaction Circuit Theory 13(1966) 82-90.

[25] A. Garcia, M.Noy and J. Teigel, Lower bounds on the number of crossing free subgraphs of K_n, Computational Geometry: Theory and Applications 16(2000), 211-221.

[26] C. Hernando, F. Hurtado, M. Marquez, M. Mora and M.Noy, Geometric tree graphs of points in convex position, Discrete Applied Mathematics 93(1999), 51-66.

[27] G. Karolyi, J. Pack and G. Toth, Ramsey type results in geometric graphs,I, Discrete and Computational Geometry 18(1997), 247-255.

[28] O. Aichholzer, F. Aurenhammer and F. Hurtado, Sequences of spanning tres and a fixed tree theorem,Computational Geometry: Theory and Applications 21(2002), 3-20.

[29] O. Aichholzer, F. Aurenhammer and F. Hurtado, Edge operations on non-crossing spanning trees. EWCG 2000, 121-125

[30] S. G. Akl, Inherently parallel geometric problems, Technical Report No. 2004-480, School of Computing, Queen's University, Kingston, Ontario, April 2004, 19 pages.

[31] S. Even, Graph Algorithms, Computer Science press, 1979.

[32] O.Ore, The Four-Color Problem, Academic Press, 1967.

[33] B. Bollobas, Graph Theory An Introductory Course, Springer-Verlag, 1979.

[34] M. Pocchiola and M. Vertger, Topologically sweeping visibility complexes via pseudotraingulations, Discrete and Computational Geometry, 16(1996), 419-453.

[35] B. Joe, Construction of three dimensional Delaunay triangulations using local transformation, Computer Aided Geometric Design, 16(1991), 419-453.

[36] F. Hurtado, M. Noy and J. Urrutia, Flipping edges in triangulation, Discrete and Computational Geometry, 22(1999), 333-346.

[37] S. Bespamyatnikh, An efficient algorithm for enumeration of triangulations, Computational Geometry, 23(2002), 271-279.