# Counting crossing – free graphs

This paper is a final project for the course "First Steps in Research for Excellent Students" as part of the B.Sc degree requirements at Tel-Aviv University under the supervision of Prof. Micha Sharir and Mr. Adam Sheffer.

By

**Gitit Kehat**

The research on this paper and the results presented in it were made possible thanks to joined efforts made by Mr. Sheffer and myself.

October 2012

# Abstract

We are interested in algorithms for counting various types of plane graphs that can be embedded over a point set $P$ of $n$ points in the plane. We present an improved analysis, showing a better running time for the algorithm of Razen and Welzl [4] for counting plane graphs, in the case where $P$ has a convex hull of at most $\frac{n+9}{2}$ vertices. In addition, we present an algorithm for counting cycle covers that uses a separator-based method, similar to the one presented in the work of Alvarez, Bringmann, Curticapean and Ray [1].


# 1. Introduction

Let $P$ be a set of $n$ points in the plane. We assume that the points in $P$ are in $general\ position$, that is, no three points are collinear. A $geometric\ graph$ on $P$ is a straight-edge graph with $P$ as its vertex set. A $crossing-free\ graph$ is a graph in which any two edges do not cross each other, means, no pair of edges shares a point other than a possible joint endpoint (a vertex of the graph). We refer to crossing-free geometric graphs as $plane\ graphs$. In this project, we are interested in counting the number of plane graph that can be embedded over $P$.

The $convex\ hull$ of $P$ is the minimal convex set containing $P$, that is, a convex polygon which has a sub-set of $P$ as its set of vertices, and all other points of $P$ are located in its interior. We refer to all points of $P$ which are located in the convex-hull interior as $inner\ vertices$. A $triangulation$ of $P$ is a plane graph on $P$ such that every inner face is a triangle, and the outer face is the exterior of the convex hull. Every triangulation is a maximal plane graph of $P$ in a way that no additional edges can be added to it without crossing existing edges.

Let $G$ be a plane graph. Then $G$ can be extended to a triangulation by adding non-crossing edges to it (repeatedly add a diagonal to every non-triangular interior face). That is, every plane graph on a set of points $P$ is contained in at least one triangulation of $P$. This property is used in [4] to count the number of plane graphs that can be embedded over $P$ with a running time that is exponentially faster than this number. We also show an improvement to the running time presented in [4] for point sets with at most $\frac{n+9}{2}$ vertices on the boundary of the convex hull.

A $crossing\ free\ cycle\ cover$ of a set of points in the plane is a plane graph, in which each vertex has a degree of 0 or 2. That is, it consists of a combination of simple cycles and isolated vertices. In Section 3, we present an algorithm for counting the number of cycle covers of a given set of points in time complexity of at most $O^*(183.33^n)$[1], which is based on the $separator$ method that is presented in [1]. This bound is far from being tight, and we hope that further analysis will yield much better results.

# 2. Exponential speed up

## 2.1 Introduction

Let $P$ be a set of $n$ points in the plane in general position. We denote by $pg(P)$ the number of plane graphs that can be embedded over $P$. We denote by $\mathcal{T}(P)$ the set of all triangulations of $P$ and by $tr(P)$ the cardinality of $\mathcal{T}(P)$. In addition, let $h$ be the number of points on the boundary of the convex hull of $P$.

An algorithm is said to $count\ with\ exponential\ speedup$ if it counts the number of elements in a set $A$ in a time exponentially faster than $|A|$. We present the exponential speed-up technique introduced by Razen and Welzl [4]. They present an algorithm that computes $pg(P)$ in a running time of $O^*(\frac{pg(P)}{\sqrt{8}^n})$, which is exponential smaller than $pg(P)$. This algorithm enumerates all triangulations of $P$, and for each triangulation $T$ counts a subset of the plane graphs that are contained in $T$, such that every plane graph of $P$ is counted exactly once. Such enumeration can be done in a time at most $O(tr(P) * loglogn)$ (Bespamyatnikh [2]). This yields the exponential speed-up, since for every set of points $P$, $tr(P)$ is exponentially smaller than $pg(P)$.

---

[1] In the notation $O^*(\cdot)$ we neglect polynomial factors.

To obtain an algorithm for counting $pg(P)$ with an exponential speedup, we still need to:

1. Show how to correctly count the number of plane graphs on $P$ by enumerating all triangulations of $P$. This will be explained in Section 2.2.
2. Prove that there is an exponential ratio between the number of triangulations and the number of plane graphs on any set $P$ in general position. This will be explained in Section 2.3.

## 2.2 Counting plane graphs by enumerating triangulations

Before we describe this part of the algorithm, let us start with a few definitions regarding triangulations and their properties.

**Definition (flippable edges).** Let $e$ be an edge in a triangulation $T$ on $P$. We say that $e$ is $flippable$ if the union of the two triangles in $T$ that contain $e$ is a $convex\ quadrilateral$ (which means $e$ is a diagonal of a convex quadrilateral, see Figure 1). The act of flipping $e$ means removing $e$ from $T$ and adding the other diagonal $e'$ of the corresponding quadrilateral, thus creating a new triangulation $T'$ on $P$. We denote by $fl(T)$ and $nfl(T)$ the number of flippable and non-flippable edges in the triangulation $T$, respectively.

**Definition (Lawson edges).** Let $e$ be a flippable edge in a triangulation $T$ on $P$. An edge $e$ is called a $Lawson\ edge$ if the sum of the two angles opposite to $e$, in the quadrilateral that is the union of the two triangles that contain $e$, is bigger than $180°$. An equivalent definition is that an edge $e$ is called a Lawson edge if the circumcircle of each of the two triangles that share $e$ also contains the other triangle (see Figure 1). Notice that this is well defined due to the general position assumption. We denote by $L(T)$ the set of Lawson edges in a triangulation $T$ and by $l(T)$ the cardinality of $L(T)$.

The $Delaunay\ triangulation\ of\ P$ is a triangulation which does not contain any Lawson edges. It exists and is unique due to the general position assumption of $P$. We recall a constrained version of this triangulation [3].

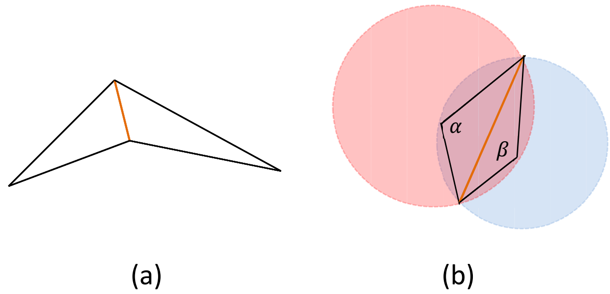


**Figure 1.** In the above quadrilaterals the emphasized orange edge is (a) non-flippable, since the quadrilateral is not convex (b) a Lawson edge since the quadrilateral lies within the intersection of the two circumcircles (c) a flippable edge but not Lawson edge since the sum of $\gamma$ and $\delta$ is less than $180°$.

**Definition (constrained Delaunay triangulation).** Let $G$ be a plane graph on $P$ (in general position). The $constrained\ Delaunay\ triangulation\ (CDT)$ $T^*(G)$ is a triangulation of $P$ such that $T^*(G)$ contains all the edges of $G$, and every other edge in $T^*(G)$ is not a Lawson edge. That is, for each triangle in $T^*(G)$, its circumcircle does not contain any other point that is visible in $G$ from one or more of the triangle vertices. Notice that $T^*(G)$ can contain Lawson edges within the constrained edges of $G$. As in the case of the Delaunay triangulation, the constrained Delaunay triangulation exists and is unique for such $P$ in general position [3]. In addition, it has been proven in [3] that given a triangulation $T \supseteq G$, the condition $L(T) \subseteq E(G) \subseteq E(T)$ is $sufficient\ and\ necessary$ for $T = T^*(G)$ to hold. This result is obtained by presenting an algorithm that receives a triangulation $T \supseteq G$ and keeps flipping Lawson edges until it reaches the constrained Delauny triangulation.

Notice that a plane graph $G$ on a point set $P$ can be contained in several triangulations of $P$. Therefore, by going over every triangulation $T$ of $P$, counting the number of plane graphs that are contained in $T$ and summing up these numbers, we might obtain a number that is much larger than $pg(P)$. The following theorem uses the above definitions to compute the exact value of $pg(P)$, by counting, for every triangulation $T$, only a specific type of subgraphs that are contained in $T$.

**Theorem 2.2.1.** Let $P$ be a set of more than 2 points in the plane in general position. Then the following holds:

$$pg(P) = \sum_{T \in \mathcal{T}(P)} 2^{M-l(T)}, \quad (1)$$

where $M$ is the number of edges in a triangulation of $P$ (all the triangulations of the same point set have the same number of edges) and recall that $l(T)$ is the number of Lawson edges in the triangulation $T$.

**Proof.** For each triangulation $T \in \mathcal{T}(P)$, we count the number of plane graphs that have $T$ as their constrained Delaunay triangulation ($CDT$). Recall that for every plane graph $G$ on $P$, $T = T^*(G)$ if and only if $L(T) \subseteq E(G) \subseteq E(T)$, so we have exactly $2^{M-l(T)}$ possible plane graphs with $T$ as their $CDT$. Recall that every plane graph $G$ has exactly one $CDT$, so every plane graph is counted exactly once in the sum (in the term that corresponds to its $CDT$). ■

## 2.3 The exponential ratio

In this section, we prove the exponential ratio between the number of plane graphs and the number of triangulations, $\frac{pg(P)}{tr(P)}$, for every point set $P$ in general position. This proof, together with the previous section, completes the analysis of the algorithm.

Dividing both sides of (1) by $tr(P)$ (the number of triangulations of $P$), we get:

$$\frac{pg(P)}{tr(P)} = \frac{1}{tr(P)} \cdot \sum_{T \in \mathcal{T}(P)} 2^{M-l(T)}. \quad (2)$$

By expectation definition, the right hand side of (2) is the expected number of subgraphs that are counted for a triangulation chosen uniformly at random from $(P)$, $\mathbb{E}\left[2^{M-l(T)}\right]$. According to Jensen's inequality and linearity of expectation we have:

$$\frac{pg(P)}{tr(P)} = \mathbb{E}\left[2^{M-l(T)}\right] \geq 2^{\mathbb{E}[M-l(T)]} = 2^{M-\mathbb{E}[l(T)]}. \quad (3)$$

**Lemma 2.3.1.** For a triangulation $T$ chosen uniformly from $\mathcal{T}(P)$, $\mathbb{E}[l(T)] = 0.5 \cdot \mathbb{E}[fl(T)]$, for any point set $P$.

**Proof sketch.** Flipping a Lawson edge in a triangulation of $P$ results in a non-Lawson edge, and vice versa. Thus, considering all instances of flippable edges in triangulations of $P$, there is a bijection between the set of Lawson edges and the set of non-Lawson edges, such that each edge is flipped to its corresponding one. This implies $\mathbb{E}[l(T)] = 0.5 \cdot \mathbb{E}[fl(T)]$. A formal proof can be found in [4]. ■

Recall that every edge in $T$ is either flippable or not, therefore $fl(T) + nfl(T) = M$. Combining this with lemma 2.3.1 and linearity of expectation, we have $\mathbb{E}[l(T)] = 0.5(M - \mathbb{E}[nfl(T)])$. Substituting the last in (3), we obtain the bound

$$\frac{pg(P)}{tr(P)} \geq 2^{0.5\cdot(M+\mathbb{E}[nfl(T)])}. \tag{4}$$

**Theorem 2.3.2.** For any set $P$ of $n$ points in general position we have:

$$pg(P) \geq \sqrt{8}^{n-1} \cdot tr(P).$$

In addition, $L(T)$ can be computed in a polynomial time for any $T$. Therefore, we can count the number of plane graphs on $P$ in $O^*(\frac{pg(P)}{\sqrt{8}^n})$ time.

**Proof.** Let $M$ be the number of edges in a triangulation $T$ of $P$. According to Euler's formula, every triangulation of $P$ has exactly $M = 3n - h - 3$ edges. By definition, every edge of the convex hull is non-flippable (since it is not a part of a convex quadrilateral), thus $nfl(T) \geq h$. Combining the above with (4) implies the asserted bound. ■

## 2.4 Our improvements

In this section, we present an improvement for the ratio of $\sqrt{8}^{n-1}$ for the case of relatively small convex hull by relying on flippability-related results from [8] and [5]. First, we recall these results together with brief proof sketches.

**Definition.** Let $\hat{v}_3$ be the expected number of inner vertices with degree 3 over all triangulations $T$ of $P$.

**Lemma 2.4.1.** In a triangulation uniformly chosen from $\mathcal{T}(P)$, $\mathbb{E}[nfl(T)] \geq 3 \cdot \hat{v}_3 + h$.

**Proof.** As shown, for example, by Souvaine, Tóth and Winslow [8], there are exactly $h + 3v_3 + 2v_{4,2} + v_{4,1}$ non-flippable edges in *any* triangulation $T$ on a set of points $P$ with $h$ convex-hull edges, where $v_{4,2}$ and $v_{4,1}$ are number of vertices of degree larger than 3 with 2 or 1 non-flippable edges, respectively. Using linearity of expectation, and ignoring $v_{4,1}$ and $v_{4,2}$, we obtain the asserted bound. ■

**Lemma 2.4.2.** $\hat{v}_3 > \frac{n-2h+9}{30}$.

**Proof.** Sharir and Sheffer [5] obtained a lower bound of $\hat{v}_3 > \frac{n-h}{30}$ for the case of triangular convex hull, using a charging scheme technique. With a new bound on the sum of all charging derived from the work in [6] and slight modification of the above proof to the case of $h > 3$, we get the required bound of $\frac{n-2h+9}{30}$. ■

**Theorem 2.4.3.** Let $P$ be a set of points in the plane in general position. Then:

$$pg(P) \geq 2^{(31n-2h-21)/20} \cdot tr(P),$$

which implies a better bound than the one presented in Theorem 2.3.2 when $h < \frac{n+9}{2}$.

**Proof.** By combining (4) with Lemmas 2.4.1 and 2.4.2, we have

$$pg(P) \geq 2^{0.5 \cdot \left(3n-3-h+h+3 \cdot \frac{n-2h+9}{30}\right)} \cdot tr(P) \geq 2^{(31n-2h-21)/20} \cdot tr(P).$$

Furthermore, there is an improvement for $h < \frac{n+9}{2}$, since $2^{(31n-2h-21)/20} > \sqrt{8}^{n-1}$ if and only if $h < \frac{n+9}{2}$. The improvement ratio is $2^{(n-2h+9)/20}$. ■

# 3. The Separator Method

In this section we present a counting technique based on the $separator$ method, as introduced by Alvarez, Bringmann, Curticapean and Ray in [1]. This technique is used in [1] for counting triangulations of a given point set in $O(3.1414^n)$ time and non-crossing spanning cycles in $n^{O(k)}$ time, where $k$ is the number of $onion\ layers$ of a point set of size $n$ (see definition in Section 3.1). In Section 3.2, we present a separator-based algorithm for counting crossing-free cycle covers of a given set of points.

**Definition (separator).** Let $P$ be a set of points in the plane and let $B$ be a polygon with all of its vertices in $P$. A set of non-crossing edges whose endpoints are the points of $P$ is called a $separator$ of $B$, if all the edges in the set lie within the interior or on the boundary of $B$, and their union divides the boundary of $B$ into at least two polygonal regions.

We will show that, to count the number of plane graphs of a specific type that can be embedded over $P$, we can choose an appropriate set $S$ of separators for the convex hull of $P$, and for each separator in $S$, recursively compute the number of graphs in each of the resulting polygonal regions. In each step of the recursion, we find a set of separators for the corresponding polygonal region. The complexity of the algorithm depends on the type of separators that we use. We first demonstrate the framework by showing a separator-based algorithm for counting the number of triangulations of a set $P$, as shown in [1].

## 3.1 Counting triangulations

In this section, we present the original application of the technique: counting triangulations of a set $P$ of $n$ points in the plane [1]. The main idea is to find a set $S$ of separators which satisfies the properties: (a) $S$ can be easily enumerated, and (b) every triangulation contains a single separator of $S$. Let $\{R_1^s, R_2^s, \dots, R_t^s\}$ be the set of polygonal regions induced by a separator $s \in S$, and $N_i^s$ be the number of triangulations of each $R_i^s$ (which can be computed recursively), then the number of triangulations containing separator $s \in S$ is $N^s = \prod_{i=1}^{t} N_i^s$ , and the number of triangulations of $P$ is $tr(P) = \sum_{s \in S} N^s$ .

In this case, we define a separator as a pair ($\Delta$,$d$), where $\Delta$ is a triangle and $d$ is a path whose vertices are in $P$. We require $\Delta$ to have at least one edge on the boundary of the polygonal region, and $d$ to have one endpoint at the vertex of $\Delta$ that is not on the boundary of the region (or, if such a vertex does not exist, at any vertex of $\Delta$), and the other endpoint at another vertex on the boundary of the region. This way, $(\Delta \cup d)$ divides the region into two sub-regions (if all the vertices of $\Delta$ are on the boundary of the region, one of the regions might be empty). We will show that these separators satisfy the separator properties stated above.

**Definition (onion layers of $\boldsymbol{P}$).** We divide the points in $P$ into $k$ *onion layers* as follows. We define the vertices on the convex-hull boundary of $P$ as the first layer. Then, we remove the vertices of the first layer, and define the next layer to be the vertices on the convex hull boundary of the remaining inner points. We repeat this process until every vertex belongs to some layer. We number each point with a unique number according to its onion layer, such that the number of a point is smaller than the number of any point in any of the following layers (an example is depicted in Figure 1.a).

**Definition ($\boldsymbol{sn-paths}$ and $\boldsymbol{sn-regions}$).** Let $T$ be a triangulation of $P$. The $sn-path$ (smallest neighbor path) of a point $p \in P$ in $T$, denoted by $sn-path_T(p)$, is the path in $T$, starting with $p$ and ending with a point on the first layer, in which every point is the smallest-neighbor in $T$ of the previous point on the path, according to our numbering. Such a path always exists, is unique, and has at most one point in each onion layer. Given an edge $(p,q)$ in $T$, the $sn-regions$ of the edge $(p,q)$ in $T$, denoted by $sn-regions\ _T(p,q)$, are the two polygonal sub-regions that are obtained by dividing a polygonal region with the union of the $sn-path_T(p)$, the $sn-path_T(q)$, and the edge $(p,q)$ (see Figure 1.b).

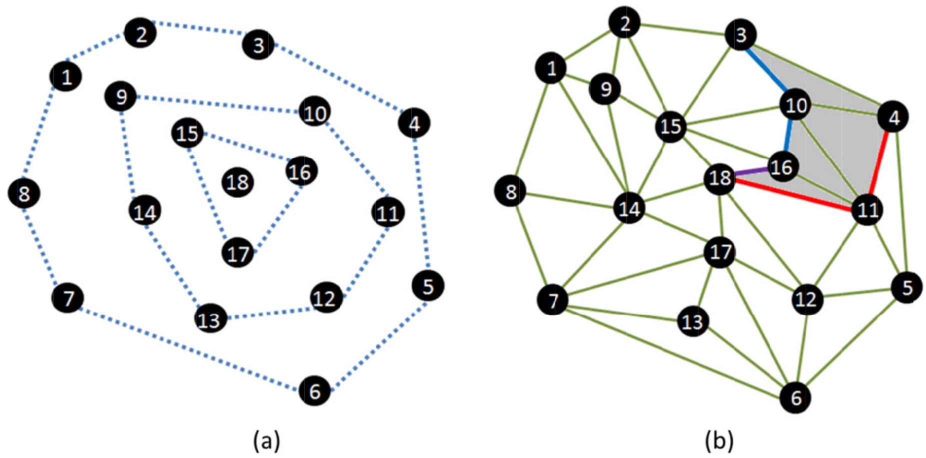


**Figure 2.** (a) A set of points with four onion layers. (b) A triangulation on the set of points. In red: $sn - path(18)$; in blue: $sn - path$(16). One of the two $sn - regions$ of edge (16,18) is emphasized in gray.

### 3.1.1 The separator choice

To define the set of separators $S$, we pick an edge $(a, b)$ on the boundary of the convex hull of $P$. In every triangulation of $P$, $(a, b)$ is incident to a $single$ triangle. We denote by $Q$ the set of all points $p \in P$ such that the triangle $apb$ does not contain any point of $P$ in its interior, and is therefore contained in a triangulation $T$ of $P$.

We define $S$ to be the set of pairs of the form $(apb, \rho)$, where $p \in Q$ and $\rho = sn - path_T(p)$ in a triangulation $T$ that contains $apb$. Every such pair partitions the respective polygonal region of $P$ into the triangle $apb$ and two polygonal sub-regions (one of which may be empty), and every triangulation contains a single separator of $S$ due to the uniqueness of $sn - path_T(p)$ in $T$. Consider a triangulation $T$ containing $apb$ and let $\rho = sn - path_T(p)$. Notice that the two polygonal sub-regions induced by such a separator are the region within $sn - regions\ _T(a, p)$ that does not contain the triangle $apb$, and the one region within $sn - regions\ _T(p, b)$ that does not contain the triangle $apb$ (for example, see Figure 3). By the definition of the $sn - paths$, for each edge $(q, r)$ on $\rho$, there is no edge $(q, u)$ where $u$ is in one of the two sub-regions, such that $u$ has a smaller number than $r$. We call this property $the\ sn - constraint$.

For each separator $s = (apb, \rho)$ (that is, a pair of a triangle and an $sn - path$), we can recursively compute the number $N^s$ of triangulations containing $s$. Let $R_1^s, R_2^s$ be the two polygonal sub-regions induced by $s$, and let $N_1^s, N_2^s$ be the number of triangulations of each sub-region, respectively. Every triangulation of $P$ that contains $s$ is composed of a triangulation of $R_1^s$ and a triangulation of $R_2^s$ (together with the triangle $apb$). Thus, $N^s = N_1^s \cdot N_2^s$ . We compute $N_1^s$ and $N_2^s$ recursively by constructing a set of separators for each of the two regions.

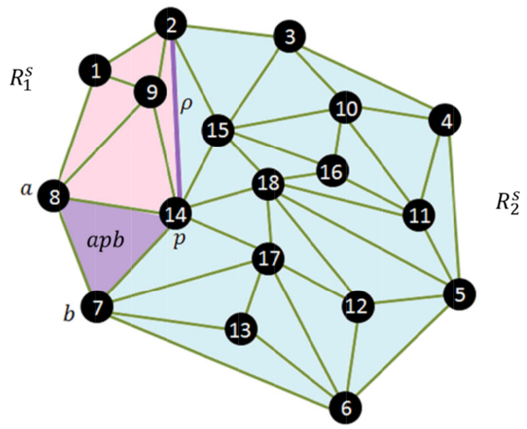


**Figure 3.** The separator partitions the convex hull into two polygonal sub-regions and the triangle $apb$.

Recall that the number of triangulations of $P$ can be computed as $tr(P) = \sum_{s \in S} N^s$. We wish to enumerate, for every $p \in Q$, all $sn - paths$ $\rho$ of $p$ in a triangulation of $P$ which contains $apb$. Moreover, we want to do this without enumerating all of the triangulations of $P$. Notice that, for a point $p$ in a region $R$ that is obtained in some step of the recursion, it is sufficient to enumerate all portions of $sn - paths$ of $p$ that lie within $R$, since the boundary of $R$ consists of parts of either the convex hull boundary or $sn - paths$ of some points. Thus, the complete set of $sn - paths$ of $p$ consist of the $sn - paths$ of $p$ that lie within $R$, or is implicitly defined by connecting the portions within $R$ with the $sn - paths$ of their corresponding endpoints on the boundary of $R$. We enumerate all portions in $R$ of $descending\ paths$ of $p$ (paths that start at $p$ and the successor of each point is on a lower onion layer). A descending path of $p$ needs not to be an $sn - path$ of $P$ in some triangulation, therefore is not necessarily a part of a separator of some triangulation. During the recursion, descending paths of $p$ that are not $sn - paths$ of $p$, will induce two regions that at least one of them does not have any triangulation that satisfies the sn-constraint.

When recurse on this kind of region, in some step we will violate the sn-constraint, and therefore the recursion will return 0.

The complexity analysis of this algorithm is done by computing an upper bound for the maximum number of overall recursive calls. Each partitioning of a region $R$ can be defined by three descending paths. Specifically, two paths that define the polygonal region $R$, and another path that divides it into two sub-regions. Once two of these three paths merge, they cannot separate again (this is because we only enumerate the portions of the third descending path within $R$, and the rest is implicitly defined) and the length of each path is bounded by the number of onion layers $k$. The analysis in [1, Section 3] implies that there are at most $O(3.1414^n)$ such triplets of paths, and thus, the running time of the algorithm is $O(3.1414^n)$.

## 3.2 Counting cycle covers

Let $P$ be a set of points in the plane in general position. Recall that a (crossing-free geometric) $cycle\ cover$ of $P$ is a plane graph $G$ that consists only of vertex-disjoint simple cycles and isolated vertices (that is, each vertex of $G$ has a degree of either 0 or 2). In this section, we present an algorithm for counting the number of cycle covers of a given set $P$. This algorithm uses separators to recursively count the number of such graphs. For now, we only obtain the bound $O^*(183.33^n)$ for the running time of the algorithm. While this is too large to be interesting, it seems far from being tight, and we hope that an improved analysis will yield a much smaller bound.

**Definition (double matching).** A $matching$ in a graph $G$ is a vertex-disjoint subset of the edges of $G$; that is, each vertex of the graph is incident to $a\ most\ one$ edge of the set. Similarly, let a $double\ matching$ in a graph $G$ be a vertex-disjoint subset of the edges of $G$, in which every vertex of the graph is incident to $at\ most\ two$ edges of the set.

Unlike matchings, the edges in a double matching can have common vertices and might form simple paths and simple cycles. We show that double matchings can be used as separators in the cycle covers counting algorithm. Since we are only interested in crossing-free cycle covers, we consider only crossing-free double matchings. In particular, we consider $crossing - free\ double\ matchings\ across\ a\ line\ l$ (and for simplicity refer to them as $double\ matchings\ across\ a\ line$), in which every edge of the matching must cross $l$.

### 3.2.1 The algorithm

Let $P$ be a set of $n$ points in the plane in general position. In the beginning of Section 3, we defined a separator of a polygon $B$ (which is embedded over a subset of $P$) as a set of non-crossing edges that divide the boundary of $B$ into at least two sub-polygons. We extend this definition by defining a $separator\ of\ a\ plane\ graph$ to be a set of non-crossing edges, whose removal separates the graph into at least two sub-graphs. For crossing-free cycle covers, the separators that we consider are sets of non-crossing edges that intersect a specific $halving\ line$ of the point set. Notice that every such separator is a double matching across a line (see Figure 4.a).

We recursively compute the number of cycle covers on $P$, denoted by $cc(P)$, such that in every step we consider a smaller subset of the points of $P$. We begin the algorithm by taking a halving line $l$ that separates the points of $P$ into two equally (or almost equally) subsets $P_L$ and $P_R$. We define a coloring of the vertices according to their degrees. Vertices that can be connected to at most two edges will be colored with blue, and vertices that can be connected to at most one or zero edges, will be colored with red and grey, respectively. At the beginning all vertices are blue, and for each sub-problem we color them accordingly. For each double matching $m$ across the line $l$, we wish to compute the number of cycle covers of $P$ that contain $m$ as a separator. We denote by $cc_{m_j}(P^i)$ the number of ways to complete the separator $m_j$ of a colored subset $P^i$, such that every vertex of $P_L^i$ and $P_R^i$ has a degree of either 0 or 2. According to this definition, the number of cycle covers that contain $m$ as a separator is $cc_m(P)$. We compute this number by recursively computing $cc_{m_L}(P_L)$ and $cc_{m_R}(P_R)$ for all respective separators, and noting that $cc_m(P) = \left[\sum_{m_L \in LM(P_L, l_L)} cc_{m_L}(P_L)\right] \cdot \left[\sum_{m_R \in LM(P_R, l_R)} cc_{m_R}(P_R)\right]$, where $LM(P, l)$ is the set of all double matchings across a line $l$ in a set $P$.

The recursion ends when a subset contains only one vertex, or when all vertices in a subset have a degree of 2 (they are all grey). The recursion returns 1 for any single vertex with an even degree or any subset of vertices with a degree of 2, and returns 0 for a single vertex with a degree of 1, since it cannot be a part of any cycle. Thus, double matchings that are not proper separators of any cycle cover will not affect the counting.

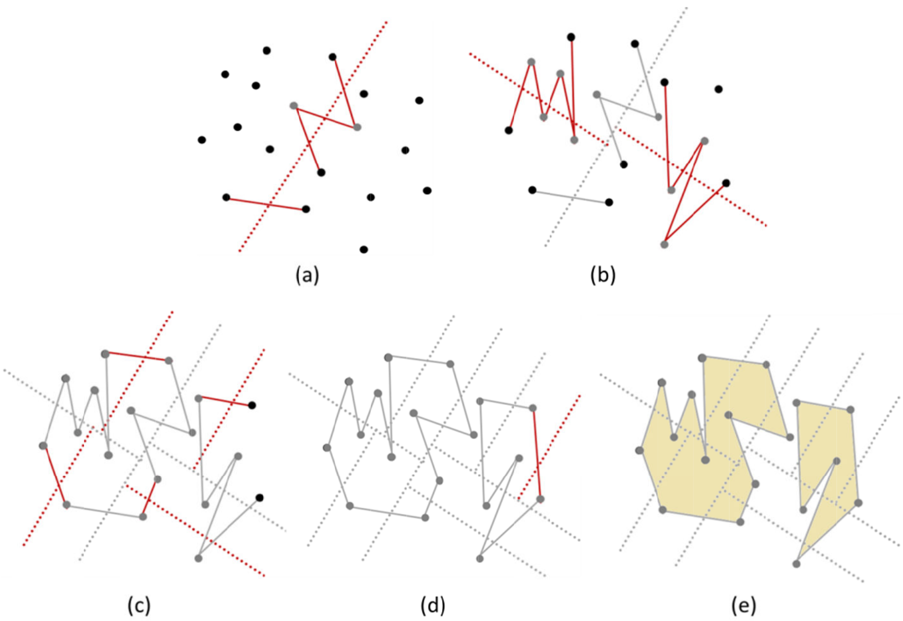

**Figure 4.** An example of a sequence of recursive calls which results in 1. In every step, a vertex with a degree of 2 is painted in grey. Notice that to form a proper cycle cover, the initial separator in (a) must be of even size. This is because the initial halving line defines a cut in the graph, which for each cycle in the cover, must cross an even number of the cycle edges.

For a given line $l$, every cycle cover of $P$ corresponds to a unique double matching across the line $l$. Thus, we can bound the number of cycle covers of $P$ as follows.

$$cc(P) = \sum_{m \in LM(P,l)} cc_m(P)\,.$$

We can enumerate over all double matchings in the following straightforward manner. First, we sort the points of $P_L$ in some order. Then, we enumerate all possibilities for connecting the first point to 0, 1 or 2 of $P_R$ points. For each of those connection possibilities, we enumerate all possibilities for connecting the second point of $P_L$ to 0, 1 or 2 of $P_R$ points, and so on. In this process, we can enumerate double matchings that are not crossing-free. However, there is an exponential number of crossing-free double matchings, and the number of edge-crossing configurations is at most $n^2$ this number. Therefore the running time of the algorithm is still dominated by the number of crossing-free double matchings.

### 3.2.2. Time complexity analysis

As mentioned above, the time complexity of the algorithm is dominated by the number of double matchings that are obtained in each step, such that every vertex has a degree of at most 2. To bound this number, it suffices to bound the number of double matchings across a halving line that a set of $n$ points can have. For this, we first recall a lemma that is proven in [7].

**Definition (perfect matching).** A $perfect\ matching$ in a graph is a matching that matches every vertex of the graph, that is, a set of edges such that every vertex of the graph is adjacent to exactly one edge of the set.

**Lemma 3.2.2.1. (Sharir and Welzl [7]).** Let $P$ be a set of $n$ points in the plane and let $l$ be a halving line of $P$. Then the number of crossing-free perfect matchings of $P$ across $l$ is $O(4^n)$.

**Proof.** Let $P_L$ and $P_R$ be the two equally sized subsets of $P$ that are induced by $l$. Let $M$ be a crossing-free perfect matching across the line $l$. The line $l$ cuts every edge $e \in M$ into a $left\ half\ edge$ and a $right\ half\ edge$. Without loss of generality, we let $l$ be the $y$-axis. Consider the vertical order in which the left half edges intersect the line $l$. While such an order may appear in more than one matching, an ordering for the left half edges, together with an ordering for the right half edges, correspond to at most one matching. Thus, the number of matchings is at most the square of the maximum number of orderings of one side. We thus consider only the points of $P_L$, the set of points to the left of $l$, and the set of all non-crossing straight segments, each starting in a point of $P_L$ and ending in a point on $l$. Notice that these segments are vertex disjoint. The increasing $y$-order of the endpoints on $l$ defines a permutation of $P_L$. We recursively bound the number of such permutations, as follows. We number the points of $P_L$ according to their order along the $x$-axis from left to right, and consider $e_1$, the left half edge starting at leftmost point $p_1$.The segment $e_1$ divides $P_L$ into two subsets $P_L^+$, $P_L^-$, containing the points of $P_L - \{p_1\}$. Rotating $e_1$ around $p_1$ results in $\frac{n}{2}$ such divisions of $P_L - \{p_1\}$. Since the graph is crossing-free, the intersection of $l$ with a left half edge whose other endpoint is above (below) $e_1$ is above (below) the intersection of

$e_1$ with $l$. Thus, the number of possible orders of intersections between $l$ and the left half edges satisfies the recursive relation

$$\Pi(\tfrac{n}{2}) \le \sum_{i=1}^{n/2} \Pi(i-1)\Pi(\tfrac{n}{2}-\mathrm{i}),\ \Pi(0) = 1.$$

This is the recursive relation of the $\frac{n}{2}$–th $Catalan\ number$, implying that the maximum number of matchings across a line is at most $C_{n/2}^2 < 4^n$. ■

**Theorem 3.2.2.2.** Let $P$ be a set of $n$ points in the plane and let $l$ be a straight line which divides $P$ into two equally sized subsets $P_L$ and $P_R$ (if $n$ is odd then the size of one set is larger by one than the size of the other). Then the maximum number of double matchings across the line $l$ is at most $13.54^n$.

**Proof.** The proof is similar to the proof of Lemma 3.2.2.1. While in a perfect matching every point is incident to exactly one edge, in a double matching each vertex may have a degree of zero, one or two. Therefore, in every step of the recursion, we divide $P_L$ into either two or three smaller subsets, or do not divide it at all, to count cases where the vertex has a degree of zero. We get the following recursive relation

$$\Pi\left(\frac{n}{2}\right) \le \Pi\left(\frac{n}{2}-1\right) + \sum\nolimits_{i} \Pi(i-1)\Pi(\frac{n}{2}-i)\ + \sum\nolimits_{i=1}^{n/2}\sum\nolimits_{j=1}^{i} \Pi(j-1)\Pi(i-j)\Pi\left(\frac{n}{2}-i\right).$$

Symmetrically, the maximum number of double matchings across a line $l$ is the square of the upper bound on the recursion relation, which, by using a numeric analysis, results to $13.54^n$. ■

A recursive call on a set of $x$ points results in at most $2 \cdot 13.54^{\frac{x}{2}}$ additional calls (each resulting in at most $2 \cdot 13.54^{\frac{x}{4}}$ calls, etc.), implying that the total number of calls is

$$13.54^n \cdot (2 \cdot 13.54^{\frac{n}{2}} \cdot (2 \cdot 13.54^{\frac{n}{4}} \cdot (\ldots = \ 13.54^{n+\frac{n}{2}+\frac{n}{4}+\cdots} \cdot 2^{logn} < n \cdot 13.54^{2n}$$
$$= O^*(183.33^n).$$

The running time of each call is polynomial, so this is also the complexity of the whole algorithm. While this is the best bound we obtained so far, it seems far from being tight.

## References

**[1]** V. Alvarez, K. Bringmann, R. Curticapean and S. Ray**,** Counting Crossing-Free Structures**.** Proc. Annual Symp on Comput. Geom. (SoCG), Chapel Hill, NC, USA ACM, (2012).

**[2]** S. Bespamyatnikh, An Efficient Algorithm for Enumeration of Triangulations. Comput. Geom. Theory Appl. 23, 271-279 (2002).

**[3]** D.T. Lee and A.K. Lin, Generalized Delaunay triangulation for planar graphs. Discrete Comput. Geom. 1, 210-217 (1986).

**[4]** A. Razen and E. Welzl, Counting Plane Graphs with Exponential Speed-Up**.** Lecture Notes in Computer Science (Springer), Vol. 6570, 36-46 (2011)**.**

**[5]** M. Sharir and A. Sheffer, Counting triangulations of planar point sets, manuscript (2010), arxiv.org/abs/0911.3352.

**[6]** M. Sharir A. Sheffer and E. Welzl, On degrees in random triangulations of point sets. J. Combinat. Theory A 118, 1979–1999 (2011).

**[7]** M. Sharir and E. Welzl, On the number of crossing-free matchings, cycles and partitions. SIAM J. Comput., 36, 695-720 (2006).

**[8]** D. L. Souvaine, Cs. D. Tóth, and A. Winslow, Simultaneously flippable edges in triangulations, *Proc. XIV Spanish Meeting on Comput. Geom.,* 137–140 (2011).