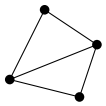


Being maximal planar is a property of an abstract graph. In contrast, a geometric graph to which no straight-line edge can be added without crossing is called a *triangulation*. Not every triangulation is maximal planar, as the example depicted to the right shows.

It is also possible to triangulate a geometric graph in linear time. But this problem is much more involved. Triangulating a single face of a geometric graph amounts to what is called "triangulating a simple polygon". This can be done in near-linear[3] time using standard techniques, and in linear time using Chazelle's famous algorithm, whose description spans a fourty pages paper [9].

Exercise 2.35. *We discussed the DCEL structure to represent plane graphs in Section 2.2.1. An alternative way to represent an embedding of a maximal planar graph is the following: For each triangle, store pointers to its three vertices and to its three neighboring triangles. Compare both approaches. Discuss different scenarios where you would prefer one over the other. In particular, analyze the space requirements of both.*

Connectivity serves as an important indicator for properties of planar graphs. Already Wagner showed that a 4-connected graph is planar if and only if it does not contain K_5 as a minor. That is, assuming 4-connectivity the second forbidden minor $K_{3,3}$ becomes "irrelevant". For subdivisions this is a different story. Independently Kelmans and Semour conjectured in the 1970s that 5-connectivity allows to consider K_5 subdivisions only. This conjecture was proven only recently[4] by Dawei He, Yan Wang, and Xingxing Yu.

Theorem 2.36 (He, Wang, and Yu [18]). *Every 5-connected nonplanar graph contains a subdivision of* K_5.

Exercise 2.37. *Give a 4-connected nonplanar graph that does not contain a subdivision of* K_5.

Another example that illustrates the importance of connectivity is the following famous theorem of Tutte that provides a sufficient condition for Hamiltonicity.

Theorem 2.38 (Tutte [32]). *Every 4-connected planar graph is Hamiltonian.*

Moreover, for a given 4-connected planar graph a Hamiltonian cycle can also be computed in linear time [10].

2.5 Compact Straight-Line Drawings

As a next step we consider geometric plane embeddings, where every edge is drawn as a straight-line segment. A classical theorem of Wagner and Fáry states that this is not a restriction to plane embeddability.

[3] $O(n \log n)$ or—using more elaborate tools—$O(n \log^* n)$ time.
[4] The result was announced in 2015 and published in 2020.

Theorem 2.39 (Fáry [13], Wagner [33]). *Every planar graph has a plane straight-line embedding.*

This is quite surprising, considering how much more freedom a simple curve allows, compared to a line segment which is completely determined by its endpoints. To further increase the level of appreciation, let us remark that a similar "straightening" is generally not possible if we fix the point set on which the vertices are to be embedded: On the one hand, Pach and Wenger [27] showed that a given planar graph G on n vertices $v_1, \dots, v_n$ and a given point set $\{p_1, \dots, p_n\} \subset \mathbb{R}^2$, one can always find a plane embedding of G such that $v_i \mapsto p_i$, for all $i \in \{1, \dots, n\}$. On the other hand, this is not possible in general with a plane *straight-line* embedding. For instance, K_4 does not admit a plane straight-line embedding on a set of points that form a convex quadrilateral, such as a rectangle. In fact, it is NP-hard to decide whether a given planar graph admits a plane straight-line embedding on a given point set [7].

Exercise 2.40. *Show the following:*

(a) For every natural number $n \geqslant 4$, there exist a planar graph G on n vertices and a set $P \subset \mathbb{R}^2$ of n points in general position (no three points are collinear) so that G does not admit a plane straight-line embedding on P.

(b) For every natural number $n \geqslant 6$, there exist a planar graph G on n vertices and a set $P \subset \mathbb{R}^2$ of n points in general position (no three points are collinear) so that (1) G does not admit a plane straight-line embedding on P; and (2) there are three points in P forming a triangle that contains all other points from P.

Exercise 2.41. *Show that for every set $P \subset \mathbb{R}^2$ of $n \geqslant 3$ in general position (no three points are collinear) the cycle on n vertices admits a plane straight-line embedding on P.*

Although Fáry-Wagner's theorem has a nice inductive proof, we do not discuss it here. Instead we will soon prove a stronger statement that implies the theorem.

A very desirable property of straight-line embeddings is that they are easy to represent: only the points/coordinates for the vertices are needed. But from an algorithmic and complexity point of view it is also important to learn the space requirement for the coordinates, since it affects the input and output size of algorithms that work on embedded graphs. While the Fáry-Wagner Theorem guarantees the existence of a plane straight-line embedding for every planar graph, it does not bound the size of the coordinates. The following strengthening provides such bounds, via an explicit algorithm that embeds (without crossing) a given planar graph on a linear size integer grid.

Theorem 2.42 (de Fraysseix, Pach, Pollack [15]). *Every planar graph on $n \geqslant 3$ vertices has a plane straight-line drawing on a $(2n-3) \times (n-1)$ integer grid. In fact, it can be computed in $O(n)$ time.*

2.5.1 Canonical Orderings

The key concept behind the algorithm is the notion of a canonical ordering, which is a vertex order that allows building the plane drawing inside out (hence canonical). Reading it backwards one may imagine a shelling or peeling order that destructs the graph from the outside. A canonical ordering also provides a succinct representation for the combinatorial embedding.

Definition 2.43. *A plane graph* G *is* **internally triangulated** *if it is biconnected and every bounded face is a (topological) triangle. We denote by* $C_\circ(G)$ *its outer cycle, that is, the cycle bounding its outer face.*

Definition 2.44. *Let* G *be an internally triangulated plane graph. A permutation* $\pi = (v_1, v_2, \ldots, v_n)$ *of* $V(G)$ *is a* **canonical ordering** *for* G *if for all* $k \in \{3, \ldots, n\}$ *we have*

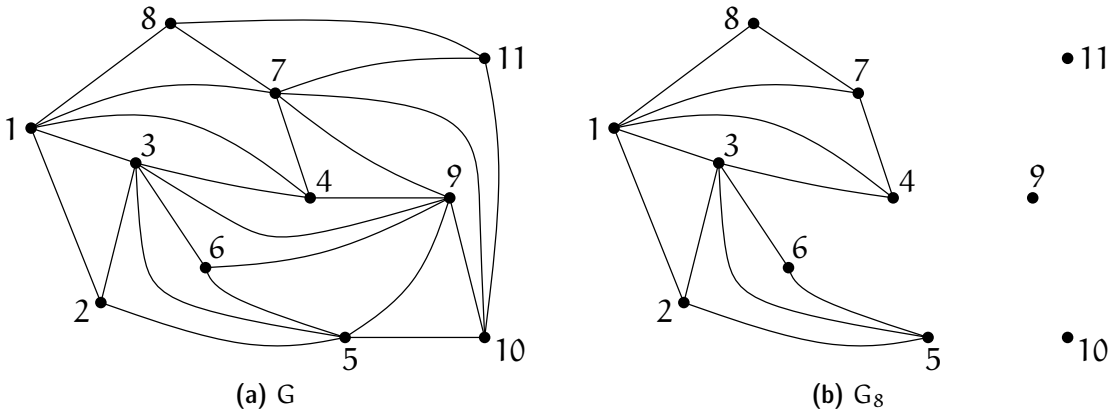
(CO1) G_k *is internally triangulated;*

(CO2) $v_1v_2 \in C_\circ(G_k)$*; and*

(CO3) v_k *is located in the outer face of* G_{k-1}*,*

where $G_k := G[\{v_1, \ldots, v_k\}]$ *is the induced drawing on the first* k *vertices.*

Figure 2.18 shows an example with canonical ordering $(1, 2, \ldots, 8)$. Note that not every permutation is a valid canonical ordering. For instance, if π chooses its first seven vertices from $\{1, 2, 3, 5, 6, 7, 8\}$, then the induced subgraph $G[\{1, 2, 3, 5, 6, 7, 8\}]$ is not biconnected since 1 is a cut vertex, thus π is not a canonical ordering. (Alternatively we may think about it backwards: Suppose we choose the initial three removals from $\{9, 10, 11\}$ as shown in Figure 2.18b, then the next removal cannot be 4 because it will leave a cut vertex in the graph.)



(a) G **(b)** G_8

Figure 2.18: *An internally triangulated plane graph with one of its canonical ordering* $(1, 2, \ldots, 8)$.

Theorem 2.45. *For every internally triangulated plane graph G and every edge v_1v_2 on its outer cycle, there exists a canonical ordering for G that starts with v_1, v_2. Moreover, such an ordering can be computed in linear time.*

Proof. Induction on n, the number of vertices. For a triangle, any ordering is valid and so the statement holds. Now consider an internally triangulated plane graph $G = (V, E)$ on $n \geqslant 4$ vertices. *Assume* we have found a vertex $v_n \in C_\circ(G) \setminus \{v_1, v_2\}$ such that the plane graph $G_{n-1} := G \setminus \{v_n\}$ is internally triangulated. (We will show later that it always exists.) Then we may apply induction on G_{n-1} and obtain a canonical ordering $(v_1, v_2, \ldots, v_{n-1})$ for G_{n-1}. The extended ordering $(v_1, v_2, \ldots, v_n)$ would satisfy (CO1)–(CO3) for $k \in \{3, \ldots, n-1\}$ by induction hypothesis, but also for $k = n$ by definition of v_n. Hence the induction would be complete, assuming the existence of v_n.

It remains to argue that v_n exists. We will show this in two steps:

(1) we can find a $v_n \in C_\circ(G) \setminus \{v_1, v_2\}$ that is not incident to a chord of $C_\circ(G)$; and

(2) such v_n automatically guarantees that $G_{n-1} := G \setminus \{v_n\}$ is internally triangulated.

First we show (1). If $C_\circ(G)$ does not have any chord, this is obvious because every cycle has at least three vertices, one of which is neither v_1 nor v_2. So suppose that $C_\circ(G)$ has a chord c. The endpoints of c split $C_\circ(G)$ into two paths, one of which does not have v_1 nor v_2 as an internal vertex. We call this path the path *associated* to c. (Such a path has at least two edges because there is always at least one vertex "behind" a chord.) Among all chords of $C_\circ(G)$ we select c such that its associated path has minimal length. Then by this choice of c its associated path together with c forms an induced cycle in G. In particular, none of the (at least one) interior vertices of the path associated to c is incident to a chord of $C_\circ(G)$ because such a chord would either cross c or it would have an associated path that is strictly shorter than the one associated to c. So we can select v_n from these vertices. By definition the path associated to c does not contain v_1 nor v_2, hence this procedure does not select either of these vertices.

Then we look at (2). The way G_{n-1} is obtained from G, every bounded face f of G_{n-1} also appears as a bounded face of G. As G is internally triangulated, f is a triangle. It remains to show that G_{n-1} is biconnected.

Consider the circular sequence of neighbors around v_n in G and break it into a linear sequence $u_1, \ldots, u_m$, for some $m \geqslant 2$, that starts and ends with the neighbors of v_n in $C_\circ(G)$. As G is internally triangulated, each of the bounded faces spanned by v_n, u_i, u_{i+1}, for $i \in \{1, \ldots, m-1\}$, is a triangle and hence $u_iu_{i+1} \in E$. The boundary of the outer face of G_{n-1} is obtained from $C_\circ(G)$ by replacing v_n with the (possibly empty) sequence $u_2, \ldots, u_{m-1}$. As v_n is not incident to a chord of $C_\circ(G)$ (and so none of $u_2, \ldots, u_{m-1}$ appeared along $C_\circ(G)$ already), the resulting sequence forms a cycle, indeed. Add a new vertex v in the outer face of G_{n-1} and connect v to every vertex of $C_\circ(G_{n-1})$ to obtain a maximal planar graph $H \supset G_{n-1}$. By Theorem 2.30 the graph H is 3-connected and so G_{n-1} is biconnected, as desired. This also completes the proof of the claim.

Regarding the runtime bound, we maintain for each vertex v whether it is on the current outer cycle and what is the number of incident chords with respect to the current

outer cycle. Given a combinatorial embedding of G, it is straighforward to initialize this information in linear time. (Every edge is considered at most twice, once for each endpoint on the outer cycle.) We also maintain an unordered list of the *eligible* vertices, that is, those vertices that are on the outer cycle and not incident to any chord. This list is straightforward to maintain: Whenever a vertex information is updated, check before and after the update whether it is eligible and correspondingly add it to or remove it from the list of eligible vertices. We store with each vertex a pointer to its position in the list (*nil* if it is not eligible currently) so that we can remove it from the list in constant time if needed.

When removing a vertex v_n from G, there are two cases: Either v_n has two neighbors u_1 and u_2 only (Figure 2.19a), in which case the edge u_1u_2 ceases to be a chord. Thus, the chord count for u_1 and u_2 has to be decremented by one. Otherwise, there are $m \geqslant 3$ neighbors $u_1, \ldots, n_m$ (Figure 2.19b) and (1) all vertices $u_2, \ldots, u_{m-1}$ are new on the outer cycle, and (2) every edge incident to u_i, for $i \in \{2, \ldots, m-1\}$, and some other vertex on the outer cycle other than u_{i-1} or u_{i+1} is a new chord. These latter changes have to be reflected in the chord counters at the vertices. So to update these counters, we inspect all edges incident to one of $u_2, \ldots, u_{m-1}$. For each such edge, we check whether the other endpoint is on the outer cycle and, if so, increment the counter.

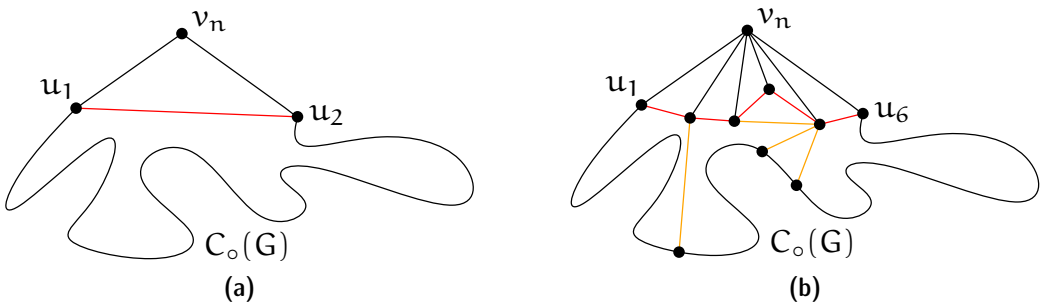


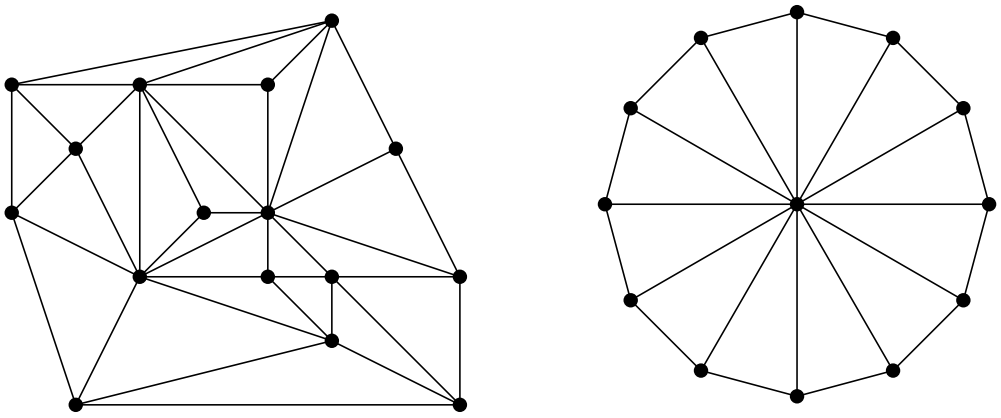
Figure 2.19: *Processing a vertex when computing a canonical ordering.*

During the course of the algorithm every vertex appears once as a new vertex on the outer cycle. At this point all incident edges (in the current graph G_i) are examined. Similarly, when a vertex v_k is removed from G_K, all edges incident to v_k in G_k are inspected; and each vertex is removed at most once. Therefore, every edge is inspected at most three times: when one of its two endpoints appears first on the outer cycle, and when the first endpoint (and therefore the edge) is removed. Altogether this takes linear time because the number of edges in G is linear by Corollary 2.5. □

Using one of the linear time planarity testing algorithms, we can obtain a combinatorial embedding for a given maximal planar graph G. As every maximal planar graph is 3-connected (Theorem 2.30), this embedding is unique (Theorem 2.26). Then, as every maximal plane graph is also internally triangulated, we can use Theorem 2.45 to provide us with a canonical ordering for (the unique embedding of) G, in overall linear time.

Corollary 2.46. *Every maximal planar graph admits a canonical ordering. Moreover, such an ordering can be computed in linear time.* □

Exercise 2.47. *(a) Compute a canonical ordering for the following internally triangulated plane graphs:*



(b) Design an infinite family of internally triangulated plane graphs on $2k$ vertices with at least $k!$ canonical orderings.

(c) Design an infinite family of internally triangulated plane graphs, along with specific choices for v_1, v_2, so that each graph in the family has a unique canonical ordering starting from v_1, v_2.

Exercise 2.48. *(a) Describe a plane graph G with n vertices that can be embedded (while preserving the outer face) in straight-line on a grid of size $(2n/3) \times (2n/3)$, but not on a smaller grid.*

(b) Can you draw G on a smaller grid if you are allowed to change the outer face?

As simple as they may appear, canonical orderings are a powerful and versatile tool to work with plane graphs. As an example, consider the following partitioning theorem.

Theorem 2.49 (Schnyder [30]). *For every maximal planar graph G on at least three vertices and every fixed face f of G, the multigraph obtained from G by doubling the (three) edges of f can be partitioned into three spanning trees.*

Exercise 2.50. *Prove Theorem 2.49.* Hint: *Fix a canonical ordering; for every vertex v_k take the edge to its first neighbor on $C_o(G_{k-1})$; argue that the edges form a spanning tree.*

Of a similar flavor is the following question.

Problem 2.51 (In memoriam Ferran Hurtado (1951–2014)).
Can every complete geometric graph on $n = 2k$ vertices (in general position) be partitioned into k plane spanning trees?

There are several positive results for special point sets [1, 5], and it is also known that there are always $\lfloor n/3 \rfloor$ edge disjoint plane spanning trees [4]. The general statement above has been refuted very recently [26]. However, it remains open if there always exists a partition into $k+1$ plane trees—or more generally, what is the minimum number of plane trees that always suffices.

2.5.2 The Shift-Algorithm

Let $(v_1, \ldots, v_n)$ be a canonical ordering of maximal planar graph G. The plan is to insert vertices in this order and extend the embedding incrementally, starting from the triangle $P(v_1) = (0,0)$, $P(v_3) = (1,1)$, $P(v_2) = (2,0)$; see Figure 2.20.

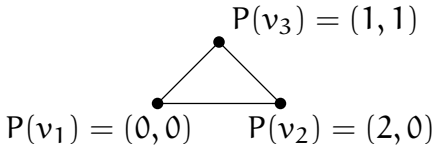


Figure 2.20: *Initialization of the shift algorithm.*

At each step, some vertices are shifted to the right, making room for the insertion of a fresh vertex. When vertex v_k is being inserted, we define a list $L(v_k)$ to memorize all vertices that need to move rigidly with v_k in the future. For the first three vertices we define $L(v_i) = \{v_i\}$, $1 \leqslant i \leqslant 3$. Once defined, a list will not change any more.

We ensure the following invariants after Step k (that is, after we have inserted v_k):

(i) We obtain a straight-line embedding of $G_k := G[\{v_1, \ldots, v_k\}]$ on the integer grid, combinatorially equivalent to the one considered in the canonical ordering. Moreover, $P(v_1) = (0,0)$ and $P(v_2) = (2k-4, 0)$.

(ii) Denote the outer cycle by $C_\circ(G_k) =: (w_1, \ldots, w_t)$ where $w_1 = v_1$ and $w_t = v_2$. The x-coordinates of $w_1, \ldots, w_t$ are strictly increasing.[5]

(iii) Each edge of $C_\circ(G_k)$ is drawn as a line segment with slope ± 1. In particular, the Manhattan distance[6] between any two points on $C_\circ(G_k)$ is even.

(iv) The lists $L(w_1), \ldots, L(w_t)$ partitions $\{v_1, \ldots, v_k\}$.

Clearly these invariants hold for G_3, embedded as described above.

Idea for Step $k+1$. We are about to place vertex v_{k+1}. Its neighbors $w_p, \ldots, w_q$ lie consecutively on $C_\circ(G_k)$ by the property of canonical ordering. Put v_{k+1} at position

[5] The notation is a bit sloppy because both t and the w_i depend on k. So in principle we should write w_i^k instead of w_i. But as the k would just make a constant appearance throughout, we omit it to avoid clutter.

[6] The *Manhattan distance* of two points (x_1, y_1) and (x_2, y_2) is $|x_2 - x_1| + |y_2 - y_1|$.

$\mu(P(w_p), P(w_q))$, where

$$\mu((x_p, y_p), (x_q, y_q)) := \left(\frac{x_p - y_p + x_q + y_q}{2}, \frac{x_p + y_p + x_q + y_q}{2}\right)$$

is the intersection between the line $y = x - x_p + y_p$ of slope 1 through (x_p, y_p) and the line $y = x_q - x + y_q$ of slope -1 through (x_q, y_q).

Proposition 2.52. *If the Manhattan distance between $P(w_p)$ and $P(w_q)$ is even, then $\mu(P(w_p), P(w_q))$ is on the integer grid.*

Proof. By (ii) we know that $x_p < x_q$. Suppose without loss of generality that $y_p \leqslant y_q$. The Manhattan distance of the two points is $d := x_q - x_p + y_q - y_p$, an even number by assumption. Adding an even number $2x_p$ to d yields the even number $x_q + x_p + y_q - y_p$, half of which is the x-coordinate of $\mu((x_p, y_p), (x_q, y_q))$. Adding an even number $2y_p$ to d yields the even number $x_q - x_p + y_q + y_p$, half of which is the y-coordinate of $\mu((x_p, y_p), (x_q, y_q))$. □

However, $\mu(P(w_p), P(w_q))$ may be unable to "see" all of $w_p, \ldots, w_q$, in case that the slope of $w_p w_{p+1}$ is 1 and/or the slope of $w_{q-1} w_q$ is -1 (Figure 2.21).

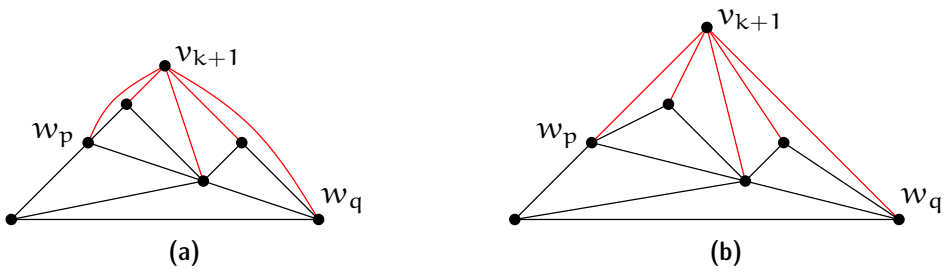


Figure 2.21: *(a) The new vertex v_{k+1} is adjacent to all of $w_p, \ldots, w_q$. If we place v_{k+1} at $\mu(P(w_p), P(w_q))$, then some edges may overlap, in case that w_{p+1} lies on the line of slope 1 through w_p or w_{q-1} lies on the line of slope -1 through w_q; (b) shifting $w_{p+1}, \ldots, w_{q-1}$ by one and $w_q, \ldots, w_t$ by two units to the right solves the problem.*

In order to resolve these problems, we shift some points to the right so that w_{p+1} no longer lies on the line of slope 1 through w_p, and that w_{q-1} no longer lies on the line of slope -1 through w_q. The actual Step $k+1$ then reads:

1. Shift $\bigcup_{i=p+1}^{q-1} L(w_i)$ to the right by one unit.
2. Shift $\bigcup_{i=q}^{t} L(w_i)$ to the right by two units.
3. $P(v_{k+1}) := \mu(P(w_p), P(w_q))$.
4. $L(v_{k+1}) := \{v_{k+1}\} \cup \bigcup_{i=p+1}^{q-1} L(w_i)$.

Next we argue that the invariants (i)–(iv) are maintained after Step $k+1$.

For (i), note that the shifting always starts from w_{p+1} onward. So $w_1 = v_1$ is never moved and stays at $P(v_1) = (0,0)$. On the other hand, we shift every vertex by two starting from (and including) w_q, hence v_2 moves two units to $P(v_2) = (2(k+1)-4, 0)$.

Also, observe that the Manhattan distance between w_p and w_q remains even because the shift increases their horizontal distance by two and leaves the y-coordinates unchanged. Therefore by Proposition 2.52 the vertex v_{k+1} is embedded on the integer grid indeed.

After shifting, the absolute slopes of the edges $w_p w_{p+1}$ and $w_{q-1} w_q$ (possibly the same edge) become < 1, and the absolute slopes of all other edges on $C_\circ(G_k)$ remain 1. In contrast, the edges $v_{k+1} w_p$ and $v_{k+1} w_q$ both have absolute slope 1, and all edges from v_{k+1} to $w_{p+1}, \ldots, w_{q-1}$ have absolute slopes > 1. Hence, for all $i \in \{p, \ldots, q\}$, the edge $v_{k+1} w_i$ intersects $C_\circ(G_k)$ in exactly one point, which is w_i. In other words, these new edges will not cross anything in G_k.

Of course, to conclude that the drawing is plane, we also need to argue that the edges originally in G_k do not clash with each other after shifting. But as this is intuitively clear, we postpone the formal argument for later. Now (i) is complete.

For (ii), clearly both the shifts and the insertion of v_{k+1} maintain the strict order along the outer cycle. For (iii), note that the edges $w_p w_{p+1}$ and $w_{q-1} w_q$ (possibly equal) are the only edges on the outer cycle $C_\circ(G_k)$ whose slope is changed. But neither edge appears on $C_\circ(G_{k+1})$ any more, as they are shadowed by the two new edges $v_{k+1} w_p$ and $v_{k+1} w_q$; the new edges have slope 1 and -1, respectively. Regarding (iv), the list $L(v_{k+1})$ by definition includes the new vertex v_{k+1} and inherits the list items from all outer cycle vertices that it shadows. So the lists on $C_\circ(G_{k+1})$ partitions $\{v_1, \ldots, v_{k+1}\}$.

So (i)–(iv) are invariants of the algorithm, indeed. Let us look at the consequences. During the entire procedure, invariants (i)(ii) and the definition of μ ensures that each point is placed on a $(2n-3) \times (n-2)$ integer grid. In fact, the final vertex v_n is always placed at $\mu(P(v_1), P(v_2)) = \mu((0,0), (2n-4, 0)) = (n-2, n-2)$ since both v_1 and v_2 are its neighbors.

Finally, we return to provide a formal argument that the "interior part" of the drawing remains plane under shifts.

Lemma 2.53. *Let G_k, $k \geqslant 3$, be straight-line embedded on grid as described by the algorithm. Assume $C_\circ(G_k) = (w_1, \ldots, w_t)$, and let $\delta_1 \leqslant \ldots \leqslant \delta_t$ be nonnegative integers. If for each i we shift $L(w_i)$ by δ_i to the right, then the resulting straight-line drawing is plane.*

Proof. Induction on k. For the base case G_3 this is obvious. Now for G_k, assume $v_k = w_\ell$, where $2 < \ell < t$. Denote its $m \geqslant 2$ neighbors as $u_1, \ldots, u_m$ where $u_1 = w_{\ell-1}$ and $u_m = w_{\ell+1}$. Then we have

$$C_\circ(G_{k-1}) = (w_1, \ldots, w_{\ell-1}, \underbrace{u_2, \ldots, u_{m-1}}_{\text{could be empty}}, w_{\ell+1}, \ldots, w_t).$$

Recall that the algorithm defines $L(v_k) = \{v_k\} \cup \bigcup_{i=1}^{m} L(u_i)$. Hence, to shift each $L(w_i)$ by δ_i is equivalent to applying the sequence

$$\Delta := (\delta_1, \ldots, \delta_{\ell-1}, \underbrace{\delta_\ell, \ldots, \delta_\ell}_{m-2 \text{ times}}, \delta_{\ell+1}, \ldots, \delta_t)$$

to G_{k-1} and then shifting v_k by δ_ℓ.

Clearly Δ is monotonically increasing, so by the inductive assumption the shifted drawing of G_{k-1} is plane. After shifting v_k by δ_ℓ, the drawing of G_k is plane: Vertex v_k moves rigidly (by exactly the same amount) with its neighbours $u_2, \ldots, u_{m-1}$ do, and the two extreme neighbours u_1 and u_m are moving relatively to the left and right, respectively. The corresponding edges cannot cross any thing during this movement. □

Linear time. The challenge in implementing the shift algorithm efficiently lies in the eponymous shift operations, which modify the x-coordinates of potentially many vertices. In fact, it is not hard to see that a naive implementation—which keeps track of all coordinates explicitly—may use quadratic time. De Fraysseix et al. described an implementation of the shift algorithm that uses $O(n \log n)$ time. Then Chrobak and Payne [11] observed how to improve the runtime to linear, using the following ideas.

Recall that v_{k+1} is placed at the coordinates

$$\begin{aligned} x &= \frac{x_p - y_p + x_q + y_q}{2}, \\ y &= \frac{(x_q - x_p) + y_p + y_q}{2}, \end{aligned} \tag{2.54}$$

and thus

$$x - x_p = \frac{(x_q - x_p) + y_q - y_p}{2}. \tag{2.55}$$

In other words, to determine the y-coordinate and the x-offset relative to the leftmost neighbour w_p, we only need the y-coordinates of w_p and w_q together with x-offset of w_q relative to w_p.

To exploit these relations, we organize the vertices in an abstract binary tree rooted at v_1, with the following structure. If we start from the root and always branch to the right, then we traverse the current outer cycle from left to right. If we branch to the left at some vertex v, then we leave the outer cycle and reach an inner vertex that was "covered" by v, that is, enclosed and thus removed from the outer face by the insertion of v. The subtree rooted at v follows the same structure recursively, so that we can walk through the historical outer cycle by branching to the right, and tap into one layer deeper by branching to the left.

More formally, for each vertex v in the tree, its *left child* is the leftmost vertex covered by the insertion of v (in the terminology from above, the vertex w_{p+1}). If no vertex is covered by v, then it left child is set to *nil*. If v is on the current outer cycle, then its *right child* is the successor of v along the current outer cycle. Otherwise, the right child

of v is the successor of v along the outer cycle at the point when both were covered together by the insertion of another vertex. If no such successor exists (for instance, for v_2 or if v is the rightmost vertex covered by some other vertex), then the right child of v is set to *nil*. See Figure 2.22 for an example.

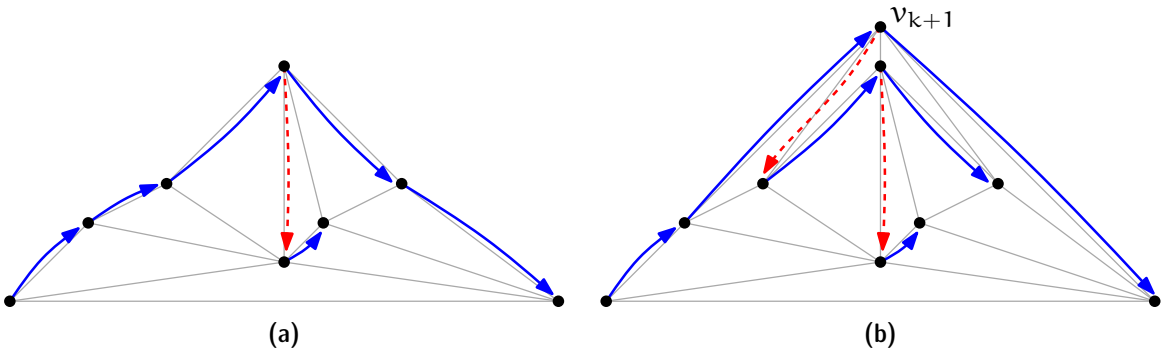


Figure 2.22: *Maintaining a binary tree representation when inserting a new vertex v_{k+1}. Red dashed arrows point to left children, blue solid arrows point to right children.*

Each tree node v also stores its x-offset $\mathrm{dx}(v)$ relative to its parent node. In this way, a whole subtree (and thus a whole set $L(\cdot)$) can be shifted virtually by changing a single offset entry at its root.

Initially, $\mathrm{dx}(v_1) = 0$, $\mathrm{dx}(v_2) = \mathrm{dx}(v_3) = 1$, $y(v_1) = y(v_2) = 0$, $y(v_3) = 1$, $\mathrm{left}(v_1) = \mathrm{left}(v_2) = \mathrm{left}(v_3) = \mathit{nil}$, $\mathrm{right}(v_1) = v_3$, $\mathrm{right}(v_2) = \mathit{nil}$, and $\mathrm{right}(v_3) = v_2$.

Inserting a vertex v_{k+1} works as follows. As before, let $w_1, \dots, w_t$ denote the vertices on the outer cycle $C_\circ(G_k)$ and $w_p, \dots, w_q$ be the neighbors of v_{k+1}.

1. Increment $\mathrm{dx}(w_{p+1})$ and $\mathrm{dx}(w_q)$ by one. *(Implement the shift.)*

2. Compute $\Delta_{pq} = \sum_{i=p+1}^{q} \mathrm{dx}(w_i)$. *(This is the total offset between w_p and w_q.)*

3. Set $\mathrm{dx}(v_{k+1}) \leftarrow \frac{1}{2}(\Delta_{pq} + y(w_q) - y(w_p))$ and $y(v_{k+1}) \leftarrow \frac{1}{2}(\Delta_{pq} + y(w_q) + y(w_p))$. *(This is exactly (2.54) and (2.55).)*

4. Set $\mathrm{right}(w_p) \leftarrow v_{k+1}$ and $\mathrm{right}(v_{k+1}) \leftarrow w_q$. *(Update the outer cycle.)*

5. If $p + 1 = q$, then set $\mathrm{left}(v_{k+1}) \leftarrow \mathit{nil}$;
   else set $\mathrm{left}(v_{k+1}) \leftarrow w_{p+1}$ and $\mathrm{right}(w_{q-1}) \leftarrow \mathit{nil}$.
   (Update $L(v_{k+1})$, the part that is covered by insertion of v_{k+1}.)

6. Set $\mathrm{dx}(w_q) \leftarrow \Delta_{pq} - \mathrm{dx}(v_{k+1})$;
   if $p + 1 \neq q$, then set $\mathrm{dx}(w_{p+1}) \leftarrow \mathrm{dx}(w_{p+1}) - \mathrm{dx}(v_{k+1})$.
   (Update the offsets according to the changes in the previous two steps.)

Observe that the only step that possibly takes more than constant time is Step 2. To analyze it, note that all vertices but the last vertex w_q for which we sum the offsets are covered by the insertion of v_{k+1}. As every vertex can be covered at most once, the overall complexity of this step during the algorithm is linear. Therefore, this first phase of the algorithm can be completed in linear time.

In a second phase, we recover the final x-coordinates from the offsets by a recursive pre-order traversal of the tree. The pseudo-code given below is to be called with the root vertex v_1 and an offset of zero. Clearly this yields a linear time algorithm overall.

```
compute_coordinate(Vertex v, Offset d) {
  if (v == nil) return;
  x(v) = dx(v) + d;
  compute_coordinate(left(v), x(v));
  compute_coordinate(right(v), x(v));
}
```

2.5.3 Remarks and Open Problems

From a geometric complexity point of view, Theorem 2.42 provides very good news for planar graphs in a similar way that the Euler Formula does from a combinatorial complexity point of view. Euler's Formula tells us that we can obtain a combinatorial representation (for instance, as a DCEL) of any plane graph using $O(n)$ space, where n is the number of vertices. Now the shift algorithm tells us that for any planar graph we can even find a geometric plane (straight-line) representation using $O(n)$ space. In addition to the combinatorial information, we only have to store $2n$ numbers from the range $\{0, 1, \ldots, 2n-4\}$.

When we make such claims regarding space complexity we implicitly assume the so-called *word RAM model*. In this model each memory cell stores a *word* of b bits, which may represent any integer in $\{0, \ldots, 2^b - 1\}$. One also assumes that b is sufficiently large, in our case $b \geqslant \log n$.

There are also different models such as the *bit complexity model*, where one is charged for every bit used to store information. In our case that would already incur an additional factor of $\log n$ for the combinatorial representation: for instance, for each halfedge we store its endpoint, which is an index from $\{1, \ldots, n\}$.

Edge lengths. Theorem 2.42 shows that planar graphs admit a plane straight-line drawing where all vertices have integer coordinates. It is an open problem whether a similar statement can be made for edge lengths.

Problem 2.56 (Harborth's Conjecture [17]). Every planar graph admits a plane straight-line drawing where all Euclidean edge lengths are integral.

Without the planarity restriction such a drawing is possible because for every $n \in \mathbb{N}$ one can find a set of n points in the plane, not all collinear, such that their distances are

all integral. In fact, such a set of points can be constructed to lie on a circle of integral radius [2]. When mapping the vertices of K_n onto such a point set, all edge lengths are integral. In the same paper it is also shown that there exists no infinite set of points in the plane so that all distances are integral, unless all of these points are collinear. Unfortunately, collinear point sets are not very useful for drawing graphs. The existence of a dense subset of the plane where all distances are rational would resolve Harborth's Conjecture. However, it is not known whether such a set exists, and in fact the suspected answer is "no".

Problem 2.57 (Erdős–Ulam Conjecture [12]). There is no dense set of points in the plane whose Euclidean distances are all rational.

Generalizing the Fáry-Wagner Theorem. As discussed earlier, not every planar graph on n vertices admits a plane straight-line embedding on every set of n points. But Theorem 2.39 states that for every planar graph G on n vertices there *exists* a set P of n points in the plane so that G admits a plane straight-line embedding on P. It is an open problem whether this statement can be generalized to hold for several graphs, in the following sense.

Problem 2.58. What is the largest number $k \in \mathbb{N}$ for which the following statement holds? For every collection of k planar graphs $G_1, \ldots, G_k$ on n vertices each, there exists a set P of n points so that G_i admits a plane straight-line embedding on P, for every $i \in \{1, \ldots, k\}$.

By Theorem 2.39 we know that the statement holds for $k = 1$. Already for $k = 2$ it is not known whether the statement holds. However, it is known that k is finite [8]. Specifically, there exists a collection of 49 planar graphs on 11 vertices each so that for every set P of 11 points in the plane at least one of these graphs does not admit a plane straight-line embedding on P [29]. Therefore we have $k \leqslant 49$.

Questions

1. *What is an embedding? What is a planar/plane graph?* Give the definitions and explain the difference between planar and plane.
2. *How many edges can a planar graph have? What is the average vertex degree in a planar graph?* Explain Euler's formula and derive your answers from it.
3. *How can plane graphs be represented on a computer?* Explain the DCEL data structure and how to work with it.
4. *How can a given plane graph be (topologically) triangulated efficiently?* Explain what it is, including the difference between topological and geometric triangulation. Give a linear time algorithm, for instance, as in Theorem 2.33.

5. *What is a combinatorial embedding? When are two combinatorial embeddings equivalent? Which graphs have a unique combinatorial plane embedding?* Give the definitions, explain and prove Whitney's Theorem.
6. *What is a canonical ordering and which graphs admit such an ordering? For a given graph, how can one find a canonical ordering efficiently?* Give the definition. State and prove Theorem 2.45.
7. *Which graphs admit a plane embedding using straight line edges? Can one bound the size of the coordinates in such a representation?* State and prove Theorem 2.42.

References

[1] Oswin Aichholzer, Thomas Hackl, Matias Korman, Marc van Kreveld, Maarten Löffler, Alexander Pilz, Bettina Speckmann, and Emo Welzl, Packing plane spanning trees and paths in complete geometric graphs. *Inform. Process. Lett.*, 124, (2017), 35–41.

[2] Norman H. Anning and Paul Erdős, Integral distances. *Bull. Amer. Math. Soc.*, 51/8, (1945), 598–600.

[3] Bruce G. Baumgart, A polyhedron representation for computer vision. In *Proc. AFIPS Natl. Comput. Conf.*, vol. 44, pp. 589–596, AFIPS Press, Alrington, Va., 1975.

[4] Ahmad Biniaz and Alfredo García, Packing plane spanning trees into a point set. *Comput. Geom. Theory Appl.*, 90, (2020), 101653.

[5] Prosenjit Bose, Ferran Hurtado, Eduardo Rivera-Campo, and David R. Wood, Partitions of complete geometric graphs into plane trees. *Comput. Geom. Theory Appl.*, 34/2, (2006), 116–125.

[6] John M. Boyer and Wendy J. Myrvold, On the cutting edge: simplified O(n) planarity by edge addition. *J. Graph Algorithms Appl.*, 8/3, (2004), 241–273.

[7] Sergio Cabello, Planar embeddability of the vertices of a graph using a fixed point set is NP-hard. *J. Graph Algorithms Appl.*, 10/2, (2006), 353–363.

[8] Jean Cardinal, Michael Hoffmann, and Vincent Kusters, On universal point sets for planar graphs. *J. Graph Algorithms Appl.*, 19/1, (2015), 529–547.

[9] Bernard Chazelle, Triangulating a simple polygon in linear time. *Discrete Comput. Geom.*, 6/5, (1991), 485–524.

[10] Norishige Chiba and Takao Nishizeki, The Hamiltonian cycle problem is linear-time solvable for 4-connected planar graphs. *J. Algorithms*, 10/2, (1989), 187–211.

[11] Marek Chrobak and Thomas H. Payne, A linear-time algorithm for drawing a planar graph on a grid. *Inform. Process. Lett.*, 54, (1995), 241–246.

[12] Paul Erdős, Ulam, the man and the mathematician. *J. Graph Theory*, 9/4, (1985), 445–449.

[13] István Fáry, On straight lines representation of planar graphs. *Acta Sci. Math. Szeged*, 11/4, (1948), 229–233.

[14] Hubert de Fraysseix, Patrice Ossona de Mendez, and Pierre Rosenstiehl, Trémaux trees and planarity. *Internat. J. Found. Comput. Sci.*, 17/5, (2006), 1017–1030.

[15] Hubert de Fraysseix, János Pach, and Richard Pollack, How to draw a planar graph on a grid. *Combinatorica*, 10/1, (1990), 41–51.

[16] Leonidas J. Guibas and Jorge Stolfi, Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. *ACM Trans. Graph.*, 4/2, (1985), 74–123.

[17] Heiko Harborth and Arnfried Kemnitz, Plane integral drawings of planar graphs. *Discrete Math.*, 236/1–3, (2001), 191–195.

[18] Dawei He, Yan Wang, and Xingxing Yu, The Kelmans-Seymour conjecture IV: A proof. *J. Combin. Theory Ser. B*, 144, (2020), 309–358.

[19] John Hopcroft and Robert E. Tarjan, Efficient planarity testing. *J. ACM*, 21/4, (1974), 549–568.

[20] Ken ichi Kawarabayashi, Yusuke Kobayashi, and Bruce Reed, The disjoint paths problem in quadratic time. *J. Combin. Theory Ser. B*, 102/2, (2012), 424–435.

[21] Lutz Kettner, *Software design in computational geometry and contour-edge based polyhedron visualization*. Ph.D. thesis, ETH Zürich, Zürich, Switzerland, 1999.

[22] Kazimierz Kuratowski, Sur le problème des courbes gauches en topologie. *Fund. Math.*, 15/1, (1930), 271–283.

[23] László Lovász, Graph minor theory. *Bull. Amer. Math. Soc.*, 43/1, (2006), 75–86.

[24] Bojan Mohar and Carsten Thomassen, *Graphs on surfaces*, Johns Hopkins University Press, Baltimore, 2001.

[25] David E. Muller and Franco P. Preparata, Finding the intersection of two convex polyhedra. *Theoret. Comput. Sci.*, 7, (1978), 217–236.

[26] Johannes Obenaus and Joachim Orthaber, Edge Partitions of Complete Geometric Graphs (Part 1). *CoRR*, abs/2108.05159.

[27] János Pach and Rephael Wenger, Embedding planar graphs at fixed vertex locations. *Graphs Combin.*, 17, (2001), 717–728.

[28] Neil Robertson and Paul Seymour, Graph Minors. XX. Wagner's Conjecture. *J. Combin. Theory Ser. B*, 92/2, (2004), 325–357.

[29] Manfred Scheucher, Hendrik Schrezenmaier, and Raphael Steiner, A Note on Universal Point Sets for Planar Graphs. *J. Graph Algorithms Appl.*, 24/3, (2020), 247–267.

[30] Walter Schnyder, Planar graphs and poset dimension. *Order*, 5, (1989), 323–343.

[31] Carsten Thomassen, Kuratowski's Theorem. *J. Graph Theory*, 5/3, (1981), 225–241.

[32] William T. Tutte, A theorem on planar graphs. *Trans. Amer. Math. Soc.*, 82/1, (1956), 99–116.

[33] Klaus Wagner, Bemerkungen zum Vierfarbenproblem. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 46, (1936), 26–32.

[34] Klaus Wagner, Über eine Eigenschaft der ebenen Komplexe. *Math. Ann.*, 114/1, (1937), 570–590.

[35] Kevin Weiler, Edge-based data structures for solid modeling in a curved surface environment. *IEEE Comput. Graph. Appl.*, 5/1, (1985), 21–40.

[36] Hassler Whitney, Congruent graphs and the connectivity of graphs. *Amer. J. Math.*, 54/1, (1932), 150–168.