Chapter 2

Plane Embeddings

Graphs can be represented in a variety of ways, for instance, as an adjacency matrix or using adjacency lists. In this chapter we explore another type of representations that are quite different in nature, namely *geometric* representations of graphs. Geometric representations are appealing because they allow to visualize a graph along with a variety of its properties in a succinct manner. There are many degrees of freedom in selecting the type of geometric objects and the details of their geometry. This freedom allows to tailor the representation to meet specific goals, such as emphasizing certain structural aspects of the graph at hand or reducing the complexity of the obtained representation.

The most common type of geometric graph representation is a *drawing*, where vertices are mapped to points and edges to curves. Making such a map injective by avoiding edge crossings is desirable, both from a mathematically aesthetic point of view and for the sake of the practical readability of the drawing. Those graphs that allow such an *embedding* into the Euclidean plane are known as *planar*. Our goal in the following is to study the interplay between abstract planar graphs and their plane embeddings. Specifically, we want to answer the following questions:

- What is the combinatorial complexity of planar graphs (number of edges and faces)?
- Under which conditions are plane embeddings unique (in a certain sense)?
- How can we represent plane embeddings (in a data structure)?
- What is the geometric complexity of plane embeddings, that is, can we bound the size of the coordinates used and the complexity of the geometric objects used to represent edges?

Most definitions we use directly extend to multigraphs. But for simplicity, we use the term "graph" throughout.

2.1 Drawings, Embeddings and Planarity

A *curve* is a set $C \subset \mathbb{R}^2$ that is of the form $\{\gamma(t) : 0 \leqslant t \leqslant 1\}$, where $\gamma : [0, 1] \to \mathbb{R}^2$ is a continuous function. The function γ is called a *parameterization* of C. The points $\gamma(0)$

and $\gamma(1)$ are the *endpoints* of the curve. For a *closed* curve, we have $\gamma(0) = \gamma(1)$. A curve is *simple*, if it admits a parameterization γ that is injective on $[0, 1]$. For a closed simple curve we allow as an exception that $\gamma(0) = \gamma(1)$. The following famous theorem describes an important property of the plane. A proof can, for instance, be found in the book of Mohar and Thomassen [24].

Theorem 2.1 (Jordan). *Any simple closed curve C partitions the plane into exactly two regions (connected open sets), each bounded by C.*

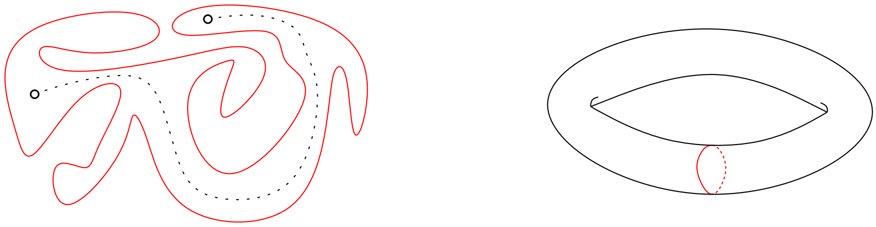


Figure 2.1: *A Jordan curve in the plane and two points in one of its faces (left); a simple closed curve that does not disconnect the torus (right).*

Observe that, for instance, on the torus there are closed curves that do not disconnect the surface (and so the theorem does not hold there).

Drawings. As a first criterion for a reasonable geometric representation of a graph, we would like to have a clear separation between different vertices and also between a vertex and nonincident edges. Formally, a *drawing* of a graph $G = (V, E)$ in the plane is a function f that assigns

- a point $f(v) \in \mathbb{R}^2$ to every vertex $v \in V$ and
- a simple curve $f(uv) : [0, 1] \to \mathbb{R}^2$ with endpoints $f(u)$ and $f(v)$ to every edge $uv \in E$,

such that

(1) f is injective on V and

(2) $f(uv) \cap f(V) = \{f(u), f(v)\}$, for every edge $uv \in E$.

A common point $f(e) \cap f(e')$ between two curves that represent distinct edges $e, e' \in E$ is called a *crossing* if it is not a common endpoint of e and e'.

For simplicity, when discussing a drawing of a graph $G = (V, E)$ it is common to treat vertices and edges as geometric objects. That is, a vertex $v \in V$ is treated as the point $f(v)$ and an edge $e \in E$ is treated as the curve $f(e)$. For instance, the last sentence of the previous paragraph may be phrased as "A common point of two edges that is not a common endpoint is called a crossing."

Often it is convenient to make additional assumptions about the interaction of edges in a drawing. For example, in a nondegenerate drawing one may demand that no three edges share a single crossing or that every pair of distinct edges intersects in at most finitely many points.

Planar vs. plane. A graph is *planar* if it admits a drawing in the plane without crossings. Such a drawing is also called a *crossing-free* drawing or a (plane) *embedding* of the graph. A planar graph together with a particular plane embedding is called a *plane* graph. Note the distinction between "planar" and "plane": the former refers to an abstract graph and indicates the possibility of an embedding, whereas the latter refers to a concrete embedding (Figure 2.2).

Figure 2.2: *A planar graph (left) and a plane embedding of it (right).*

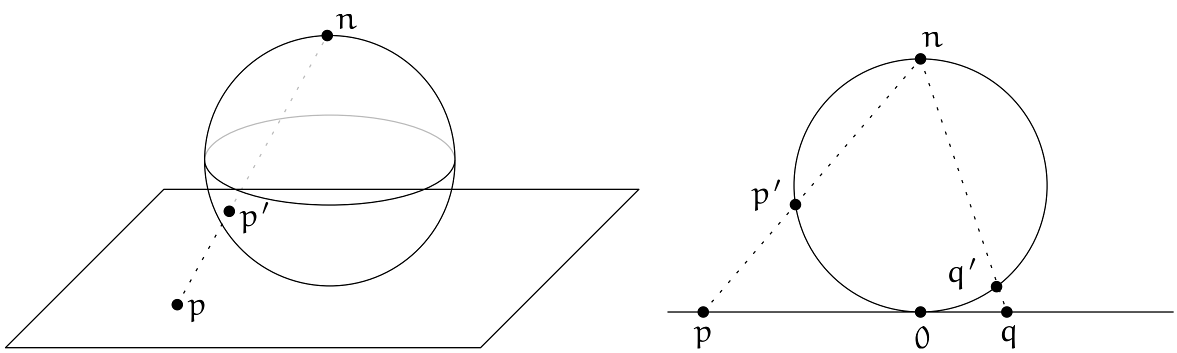
A *geometric graph* is a graph together with a drawing, in which all edges are realized as straight-line segments. Note that such a drawing is completely defined by the mapping for the vertices. A plane geometric graph is also called a *plane straight-line graph* (PSLG). In contrast, a plane graph in which the edges may form arbitrary simple curves is called a *topological plane graph*.

The *faces* of a plane graph are the maximally connected regions of the plane that do not contain any point used by the embedding (as the image of a vertex or an edge). Each embedding of a finite graph has exactly one *unbounded face*, also called *outer* or *infinite* face. Using stereographic projection, it is not hard to show that the role of the unbounded face is not as special as it may seem at first glance.

Theorem 2.2. *If a graph* G *has a plane embedding in which some face is bounded by the cycle* $(v_1, \ldots, v_k)$*, then* G *also has a plane embedding in which the unbounded face is bounded by the cycle* $(v_1, \ldots, v_k)$*.*

Proof. **(Sketch)** Take a plane embedding Γ of G and map it to the sphere using *stereographic projection*: Imagine $\mathbb{R}^2$ being the x/y-plane in $\mathbb{R}^3$ and place a unit sphere S such that its south pole touches the origin. We obtain a bijective continuous map between $\mathbb{R}^2$ and $S \setminus \{n\}$, where n is the north pole of S, as follows: A point $p \in \mathbb{R}^2$ is mapped to the point p' that is the intersection of the line through p and n with S, see Figure 2.3.

Consider the resulting embedding Γ' of G on S: The infinite face of Γ corresponds to the face of Γ' that contains the north pole n of S. Now rotate the embedding Γ' on S

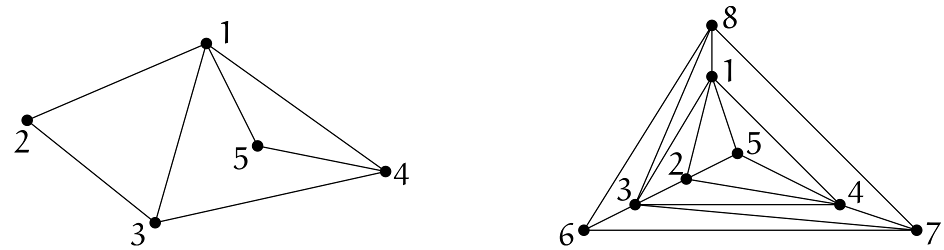


(a) Three-dimensional view. (b) Cross-section view.

Figure 2.3: *Stereographic projection.*

such that the desired face contains n. Mapping back to the plane using stereographic projection results in an embedding in which the desired face is the outer face. □

Exercise 2.3. *Consider the plane graphs depicted in Figure 2.4. For both graphs give a plane embedding in which the cycle* $1, 2, 3$ *bounds the outer face.*



(a) (b)

Figure 2.4: *Make* $1, 2, 3$ *bound the outer face.*

Duality. Every plane graph G has a *dual* G^*, whose vertices are the faces of G and two are connected by an edge in G^*, if and only if they have a common edge in G. In general, G^* is a multigraph (may contain loops and multiple edges) and it depends on the embedding. That is, an abstract planar graph G may have several nonisomorphic duals; see Figure 2.5 for an example. If G is a connected plane graph, then $(G^*)^* = G$. We will see later in Section 2.3 that the dual of a 3-connected planar graph is unique (up to isomorphism).

The Euler Formula and its ramifications. One of the most important tools for planar graphs (and more generally, graphs embedded on a surface) is the Euler–Poincaré Formula.

Theorem 2.4 (Euler's Formula). *For every connected plane graph with* n *vertices,* e *edges, and* f *faces, we have* $n - e + f = 2$.

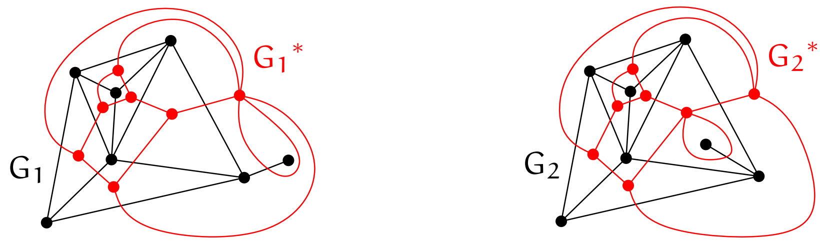


Figure 2.5: *Two plane drawings G_1 and G_2 of the same abstract planar graph and their duals $G_1{}^*$ and $G_2{}^*$ with $G_1{}^* \not\simeq G_2{}^*$. (To see this, for instance, count the number of vertices of degree greater than three.)*

Proof. Let G be a connected plane graph with n vertices, e edges, and f faces. We prove the statement by induction on $e-n$. As G is connected, it contains a spanning tree T, which has $n-1$ edges. In the base case, we have $G = T$ and $e - n = 1$. A plane tree has exactly one (unbounded) face, and so $n - e + f = n - (n - 1) + 1 = 2$, as claimed.

In the general case, fix a spanning tree T of G, pick an arbitrary edge e of $G \setminus T$, and consider the graph $G^- = G \setminus e$. By construction G^- has n vertices and $e - 1$ edges. We claim that it has $f - 1$ faces. To see this observe that G^- is connected because $T \subset G^-$. In particular, the endpoints of e are connected by a path in G^-, which together with e forms a cycle in G. So in G any two points sufficiently close to but on different sides of e are in different faces, whereas these points are in the same face of G^-. It follows that G^- has $f - 1$ faces, as claimed. Then by the inductive assumption, applied to G^-, we have $2 = n - (e - 1) + (f - 1) = n - e + f$, which proves the theorem. □

In particular, this shows that for any planar graph the number of faces is the same in every plane embedding. In other words the number of faces is an invariant of an abstract planar graph. It also follows (stated below as a corollary) that planar graphs are *sparse*, that is, they have a linear number of edges (and faces) only. So the asymptotic complexity of a planar graph is already determined by its number of vertices.

Corollary 2.5. *A simple planar graph on $n \geqslant 3$ vertices has at most $3n - 6$ edges and at most $2n - 4$ faces.*

Proof. Without loss of generality we may assume that G is connected. (If not, add edges between components of G until the graph is connected. The number of faces remains unchanged and the number of edges only increases.) The statement is easily checked for $n = 3$, where G is either a triangle or a path and, therefore, has no more than $3 \cdot 3 - 6 = 3$ edges and no more than $2 \cdot 3 - 4 = 2$ faces. So consider a simple planar graph G on $n \geqslant 4$ vertices. Consider a plane drawing of G and denote by E the set of edges and by F the set of faces of G. Let

$$X = \{(e, f) \in E \times F : e \text{ bounds } f\}$$

denote the set of incident edge-face pairs. We count X in two different ways.

First note that each edge bounds at most two faces and so $|X| \leqslant 2 \cdot |E|$.

Second note that in a simple connected planar graph on four or more vertices every face is bounded by at least three vertices: Every bounded face needs at least three edges to be enclosed and if there is no cycle on the boundary of the unbounded face, then—given that G is connected—G must be a tree on four or more vertices and so it has at least three edges, all of which bound the unbounded face. Therefore $|X| \geqslant 3 \cdot |F|$.

Using Euler's Formula we conclude that

$$\begin{aligned} 4 &= 2n - 2|E| + 2|F| \leqslant 2n - 3|F| + 2|F| = 2n - |F| \text{ and} \\ 6 &= 3n - 3|E| + 3|F| \leqslant 3n - 3|E| + 2|E| = 3n - |E|, \end{aligned}$$

which yields the claimed bounds. □

It also follows that the degree of a "typical" vertex in a planar graph is a small constant. There exist several variations of this statement, a few more of which we will encounter during this course.

Corollary 2.6. *The average vertex degree in a simple planar graph is less than six.*

Exercise 2.7. *Prove Corollary 2.6.*

Exercise 2.8. *Show that neither K_5 (the complete graph on five vertices) nor $K_{3,3}$ (the complete bipartite graph where both classes have three vertices) is planar.*

Exercise 2.9. *Let P be a set of $n \geqslant 3$ points in the plane such that the distance between any pair of points is at least one. Show that there are at most $3n-6$ pairs of points in P at distance exactly one.*

Characterizing planarity. The classical theorems of Kuratowski and Wagner provide a characterization of planar graphs in terms of forbidden substructures. A *subdivision* of a graph $G = (V, E)$ is a graph that is obtained from G by replacing each edge with a path.

Theorem 2.10 (Kuratowski [22, 31]). *A graph is planar if and only if it does not contain a subdivision of $K_{3,3}$ or K_5.*

A *minor* of a graph $G = (V, E)$ is a graph that is obtained from G using zero or more edge contractions, edge deletions, and/or vertex deletions.

Theorem 2.11 (Wagner [34]). *A graph is planar if and only if it does not contain $K_{3,3}$ or K_5 as a minor.*

In some sense, Wagner's Theorem is a special instance[1] of a much more general theorem.

Theorem 2.12 (Graph Minor Theorem, Robertson/Seymour [28]). *Every minor-closed family of graphs can be described in terms of a finite set of forbidden minors.*

[1] It is more than just a special instance because it also specifies the forbidden minors explicitly.

Being *minor-closed* means that for every graph from the family also all of its minors belong to the family. For instance, the family of planar graphs is minor-closed because planarity is preserved under removal of edges and vertices and under edge contractions.

Exercise 2.13. *The family of* 1*-planar graphs consists of all graphs that admit a drawing in the plane in which every edge has at most one crossing. Prove or disprove: The family of* 1*-planar graphs is minor-closed.*

The Graph Minor Theorem is a celebrated result that was established by Robertson and Seymour in a series of twenty papers, see also the survey by Lovász [23]. They also describe an $O(n^3)$ algorithm (with horrendous constants, though) to decide whether a graph on n vertices contains a fixed (constant-size) minor. Later, Kawarabayashi et al. [20] showed that this problem can be solved in $O(n^2)$ time. As a consequence, every minor-closed property can be decided in polynomial time.

Unfortunately, the result is nonconstructive in the sense that in general we do not know how to obtain the set of forbidden minors for a given family/property. For instance, for the family of toroidal graphs (graphs that can be embedded without crossings on the torus) more than 16′000 forbidden minors are known, and we do not know how many there are in total. So while we know that there exists a quadratic time algorithm to test membership for minor-closed families, we have no idea what such an algorithm looks like in general.

Graph families other than planar graphs for which the forbidden minors are known include forests (K_3) and outerplanar graphs ($K_{2,3}$ and K_4). A graph is *outerplanar* if it admits a plane embedding such that all vertices appear on the outer face (Figure 2.6).

Figure 2.6: *An outerplanar graph (left) and a plane embedding of it in which all vertices are incident to the outer face (right).*

Exercise 2.14. *(a) Give an example of a* 6*-connected planar graph or argue that no such graph exists.*

(b) Give an example of a 5*-connected planar graph or argue that no such graph exists.*

(c) Give an example of a 3*-connected outerplanar graph or argue that no such graph exists.*

Planarity testing. For planar graphs we do not have to contend ourselves with a cubic-time algorithm. Rather, it can be decided in linear time whether or not a given abstract graph is planar. In fact, there exist a number of different linear time algorithms, all of which—from a very high-level point of view—can be regarded as an annotated depth-first-search. The first such algorithm was described by Hopcroft and Tarjan [19], while the current state-of-the-art is probably among the "path searching" method by Boyer and Myrwold [6] and the "LR-partition" method by de Fraysseix et al. [14]. Although the overall idea in all these approaches is easy to convey, there are many technical details, which make an in-depth discussion rather painful to go through.

2.2 Graph Representations

There are two standard representations for an abstract graph $G = (V, E)$ on $n = |V|$ vertices. For the *adjacency matrix* representation we consider the vertices to be ordered as $V = \{v_1, \ldots, v_n\}$. The adjacency matrix of an undirected graph is a symmetric $n \times n$-matrix $A = (a_{ij})_{1 \leqslant i,j \leqslant n}$ where $a_{ij} = a_{ji} = 1$, if $\{i, j\} \in E$, and $a_{ij} = a_{ji} = 0$, otherwise. Storing such a matrix explicitly requires $\Omega(n^2)$ space, and allows to test in constant time whether or not two given vertices are adjacent.

In an *adjacency list* representation, we store for each vertex a list of its neighbors in G. This requires only $O(n + |E|)$ storage, which is better than for the adjacency matrix in case that $|E| = o(n^2)$. On the other hand, the adjacency test for two given vertices is not a constant-time operation, because it requires a search in one of the lists. Depending on the representation of these lists, such a search takes $O(d)$ time (unsorted list) or $O(\log d)$ time (sorted random-access representation, such as a balanced search tree), where d is the minimum degree of the two vertices.

Both representations have their merits. The choice of which one to use (if any) typically depends on what one wants to do with the graph. When dealing with embedded graphs, however, additional information concerning the embedding is needed beyond the pure incidence structure of the graph. The next section discusses a standard data structure to represent embedded graphs.

2.2.1 The Doubly-Connected Edge List

The *doubly-connected edge list* (DCEL) is a data structure to represent a plane graph in such a way that it is easy to traverse and to manipulate. In order to avoid unnecessary complications, let us discuss only connected graphs here that contain at least two vertices. It is not hard to extend the data structure to cover all plane graphs. For simplicity we also assume that we deal with a straight-line embedding and so the geometry of edges is defined by the mapping of their endpoints already. For more general embeddings, the geometric description of edges has to be stored in addition.

The main building block of a DCEL is a list of *halfedges*. Every actual edge is represented by two halfedges going in opposite direction, and these are called *twins*, see Figure 2.7. Along the boundary of each face, halfedges are oriented counterclockwise.

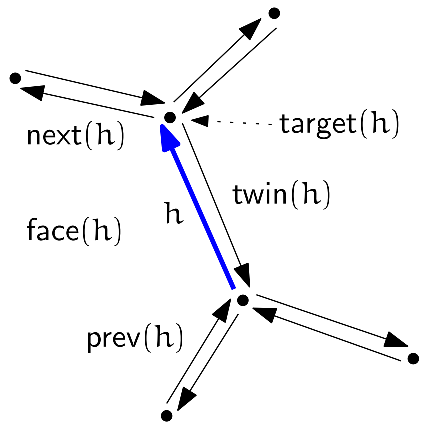


Figure 2.7: *A halfedge in a DCEL.*

A DCEL stores a list of halfedges, a list of vertices, and a list of faces. These lists are unordered but interconnected by various pointers. A vertex v stores a pointer halfedge(v) to an arbitrary halfedge originating from v. Every vertex also knows its coordinates, that is, the point point(v) it is mapped to in the represented embedding. A face f stores a pointer halfedge(f) to an arbitrary halfedge within the face. A halfedge h stores *five* pointers:

- a pointer target(h) to its target vertex,
- a pointer face(h) to the incident face,
- a pointer twin(h) to its twin halfedge,
- a pointer next(h) to the halfedge following h along the boundary of face(h), and
- a pointer prev(h) to the halfedge preceding h along the boundary of face(h).

A constant amount of information is stored for every vertex, (half-)edge, and face of the graph. Therefore the whole DCEL needs storage proportional to $|V| + |E| + |F|$, which is $O(n)$ for a plane graph with n vertices by Corollary 2.5.

This information is sufficient for most tasks. For example, traversing all edges around a face f can be done as follows:

```
s ← halfedge(f)
h ← s
do
    something with h
    h ← next(h)
while h ≠ s
```

Exercise 2.15. *Give pseudocode to traverse all edges incident to a given vertex v of a DCEL.*

Exercise 2.16. *Why is the previous halfedge* prev(·) *stored explicitly and the source vertex of a halfedge is not?*

2.2.2 Manipulating a DCEL

In many applications, plane graphs appear not just as static objects but rather they evolve over the course of an algorithm. Therefore the data structure used to represent the graph must allow for efficient update operations to change it.

First of all, we need to be able to generate new vertices, edges, and faces, to be added to the corresponding list within the DCEL and—symmetrically—the ability to delete an existing entity. Then it should be easy to add a new vertex v to the graph within some face f. As we maintain a connected graph, we better link the new vertex to somewhere, say, to an existing vertex u. For such a connection to be possible, we require that the open line segment uv lies completely in f.

Of course, two halfedges are to be added connecting u and v. But where exactly? Given that from a vertex and from a face only some arbitrary halfedge is directly accessible, it turns out convenient to use a halfedge in the interface. Let h denote the halfedge incident to f for which $\text{target}(h) = u$. Our operation then becomes (see also Figure 2.8)

add-vertex-at(v, h)
Precondition: the open line segment $\overline{\text{point}(v)\text{point}(u)}$, where $u := \text{target}(h)$, lies completely in $f := \text{face}(h)$.
Postcondition: a new vertex v has been inserted into f, connected by an edge to u.

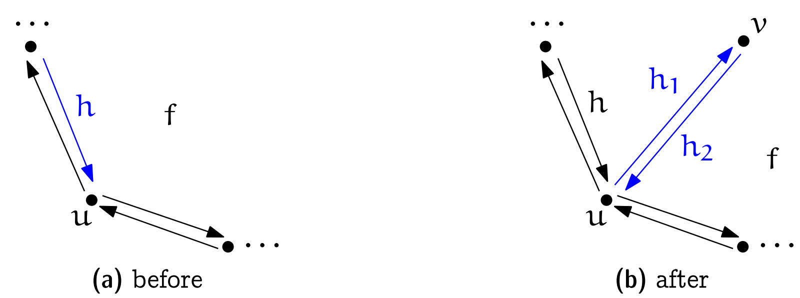


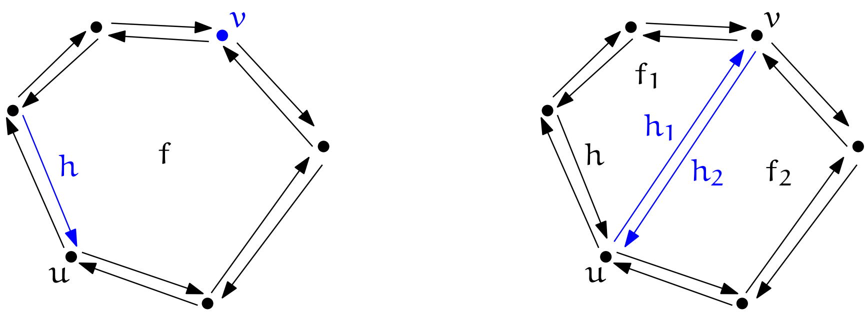
Figure 2.8: *Add a new vertex connected to an existing vertex* u.

and it can be realized by manipulating a constant number of pointers as follows.

add-vertex-at(v, h) {
  $h_1 \leftarrow$ a new halfedge
  $h_2 \leftarrow$ a new halfedge
  halfedge$(v) \leftarrow h_2$
  twin$(h_1) \leftarrow h_2$

$\text{twin}(h_2) \leftarrow h_1$
$\text{target}(h_1) \leftarrow v$
$\text{target}(h_2) \leftarrow u$
$\text{face}(h_1) \leftarrow f$
$\text{face}(h_2) \leftarrow f$
$\text{next}(h_1) \leftarrow h_2$
$\text{next}(h_2) \leftarrow \text{next}(h)$
$\text{prev}(h_1) \leftarrow h$
$\text{prev}(h_2) \leftarrow h_1$
$\text{next}(h) \leftarrow h_1$
$\text{prev}(\text{next}(h_2)) \leftarrow h_2$
}

Similarly, it should be possible to add an edge between two existing vertices u and v, provided the open line segment uv lies completely within a face f of the graph, see Figure 2.9. Since such an edge insertion splits f into two faces, the operation is called *split-face*. Again we use the halfedge h that is incident to f and for which $\text{target}(h) = u$.



(a) before **(b)** after

Figure 2.9: *Split a face by an edge uv.*

Our operation becomes then

split-face(h, v)
Precondition: v is incident to $f := \text{face}(h)$ but not adjacent to $u := \text{target}(h)$.
The open line segment $\overline{\text{point}(v)\text{point}(u)}$ lies completely in f.
Postcondition: f has been split by a new edge uv.

The implementation is slightly more complicated compared to add-vertex-at above, because the face f is destroyed and so we have to update the face information of all incident halfedges. In particular, this is not a constant time operation, but its time complexity is proportional to the size of f.

split-face(h, v) {
$f_1 \leftarrow$ a new face

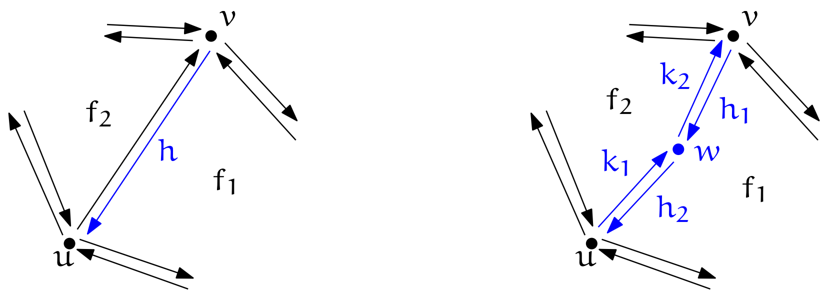
```
    f_2 ← a new face
    h_1 ← a new halfedge
    h_2 ← a new halfedge
    halfedge(f_1) ← h_1
    halfedge(f_2) ← h_2
    twin(h_1) ← h_2
    twin(h_2) ← h_1
    target(h_1) ← v
    target(h_2) ← u
    next(h_2) ← next(h)
    prev(next(h_2)) ← h_2
    prev(h_1) ← h
    next(h) ← h_1
    i ← h_2
    loop
        face(i) ← f_2
        if target(i) = v break the loop
        i ← next(i)
    endloop
    next(h_1) ← next(i)
    prev(next(h_1)) ← h_1
    next(i) ← h_2
    prev(h_2) ← i
    i ← h_1
    do
        face(i) ← f_1
        i ← next(i)
    until target(i) = u
    delete the face f
}
```

In a similar fashion one can realize the inverse operation join-face(h) that removes the edge (represented by the halfedge) h, thereby joining the faces face(h) and face(twin(h)).

It is easy to see that every connected plane graph on at least two vertices can be constructed using the operations add-vertex-at and split-face, starting from an embedding of K_2 (two vertices connected by an edge).

Exercise 2.17. *Give pseudocode for the operation join-face*(h). *Also specify preconditions, if needed.*

Exercise 2.18. *Give pseudocode for the operation split-edge*(h), *that splits the edge (represented by the halfedge) h into two by a new vertex w, see Figure 2.10.*



(a) before **(b)** after

Figure 2.10: *Split an edge by a new vertex.*

2.2.3 Graphs with Unbounded Edges

In some cases it is convenient to consider plane graphs, in which some edges are not mapped to a line segment but to an unbounded curve, such as a ray. This setting is not really much different from the one we studied before, except that one vertex is placed "at infinity". One way to think of it is in terms of *stereographic projection* (see the proof of Theorem 2.2). The further away a point in $\mathbb{R}^2$ is from the origin, the closer its image on the sphere S gets to the north pole n of S. But there is no way to reach n except in the limit. Therefore, we can imagine drawing the graph on S instead of in $\mathbb{R}^2$ and putting the "infinite vertex" at n.

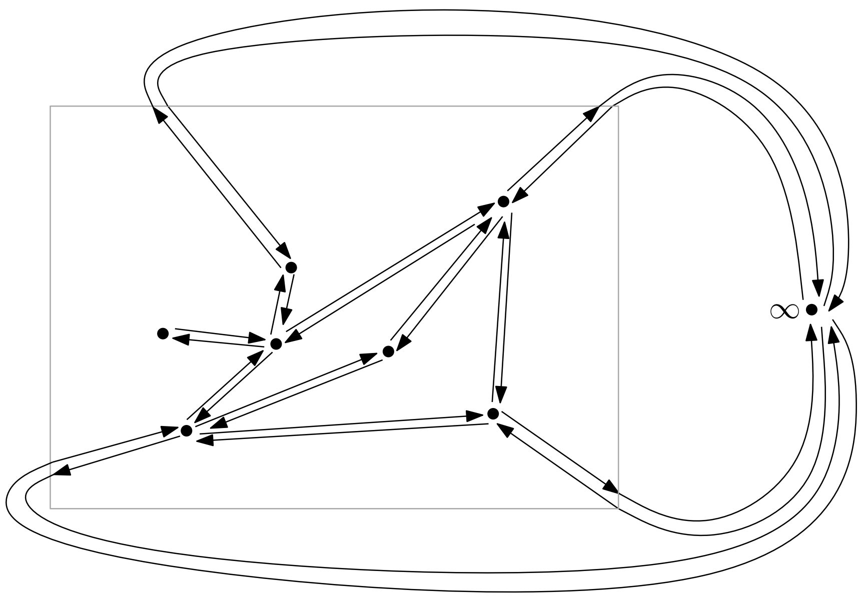


Figure 2.11: *A DCEL with unbounded edges. Usually, we will not show the infinite vertex and draw all edges as straight-line segments. This yields a geometric drawing, like the one within the gray box.*

All this is just for the sake of a proper geometric interpretation. As far as a DCEL representation of such a graph is concerned, there is no need to consider spheres or, in fact, anything beyond what we have discussed before. The only difference to the case with all finite edges is that there is this special infinite vertex, which does not have any point/coordinates associated to it. But other than that, the infinite vertex is treated in exactly the same way as the finite vertices: it has in– and outgoing halfedges along which the unbounded faces can be traversed (Figure 2.11).

Remarks. It is actually not so easy to point exactly to where the DCEL data structure originates from. Often Muller and Preparata [25] are credited, but while they use the term DCEL, the data structure they describe is different from what we discussed above and from what people usually consider a DCEL nowadays. Overall, there are a large number of variants of this data structure, which appear under the names *winged edge* data structure [3], *halfedge* data structure [35], or *quad-edge* data structure [16]. Kettner [21] provides a comparison of all these and some additional references.

2.2.4 Combinatorial Embeddings

The DCEL data structure discussed in the previous section provides a fully fleshed-out representation of what is called a *combinatorial embedding*. From a mathematical point of view this can be regarded an equivalence relation on embeddings: Two embeddings are equivalent if their face boundaries—regarded as circular sequences of edges (or vertices) in counterclockwise order—are the same (as sets) up to a global change of orientation (reversing the order of all sequences simultaneously). For instance, the faces of the plane graphs shown in Figure 2.12a are (each face is described as a circular sequence of vertices)

$$\begin{aligned}(a) &: \{(1,2,3),(1,3,6,4,5,4),(1,4,6,3,2)\},\\ (b) &: \{(1,2,3,6,4,5,4),(1,3,2),(1,4,6,3)\}, \text{and}\\ (c) &: \{(1,4,5,4,6,3),(1,3,2),(1,2,3,6,4)\}.\end{aligned}$$

Note that a vertex can appear several times along the boundary of a face (if it is a cut-vertex). Clearly (b) is not equivalent to (a) nor (c), because it is the only graph that contains a face bounded by seven vertices. However, (a) and (c) turn out to be equivalent: after reverting orientations f_1 takes the role of h_2, f_2 takes the role of h_1, and f_3 takes the role of h_3.

Exercise 2.19. *Let* G *be a planar graph with vertex set* $\{1,\ldots,9\}$*. Try to find an embedding corresponding to the following list of circular sequences of faces:*

(a) $\{(1,4,5,6,3),(1,3,6,2),(1,2,6,7,8,9,7,6,5),(7,9,8),(1,5,4)\}$

(b) $\{(1,4,5,6,3),(1,3,6,2),(1,2,6,7,8,9,7,6,5),(7,9,8),(1,4,5)\}$

In a dual interpretation one can just as well define equivalence in terms of the cyclic order of neighbors around all vertices. In this form, a compact way to describe a combinatorial embedding is as a so-called *rotation system* that consists of a permutation π

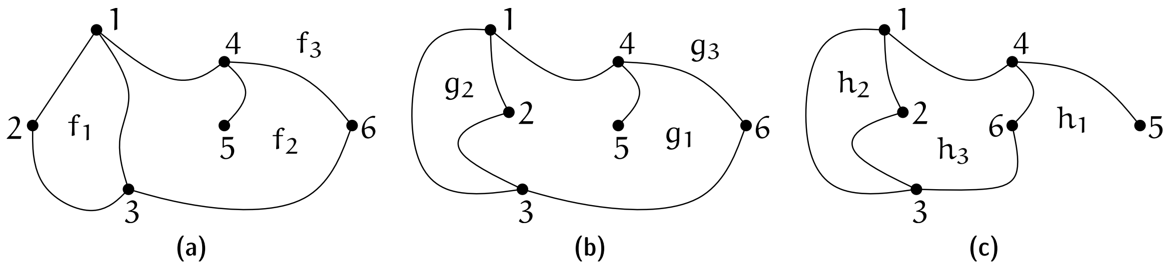


Figure 2.12: *Equivalent embeddings?*

and an involution ρ, both of which are defined on the set of halfedges (in this context often called *darts* or *flags*) of the embedding. The orbits of π correspond to the vertices, as they iterate over the incident halfedges. The involution ρ maps each halfedge to its twin.

Many people prefer this dual view because one does not have to discuss the issue of vertices or edges that appear several times on the boundary of a face. The following lemma shows that such an issue does not arise when dealing with biconnected graphs.

Lemma 2.20. *In a biconnected plane graph every face is bounded by a cycle.*

We leave the proof as an exercise. Intuitively the statement is probably clear. But we believe it is instructive to think about how to make a formal argument. An easy consequence is the following corollary, whose proof we also leave as an exercise.

Corollary 2.21. *In a 3-connected plane graph the neighbors of a vertex lie on a cycle.*

Note that the statement does not read "form a cycle" but rather "lie on a cycle".

Exercise 2.22. *Prove Lemma 2.20 and Corollary 2.21.*

2.3 Unique Embeddings

We have seen in Lemma 2.20 that all faces in biconnected plane graphs are bounded by cycles. Conversely one might wonder which cycles of a planar graph G bound a face in *some* plane embedding of G. Such a cycle is called a *facial cycle* (Figure 2.13).

Exercise 2.23. *Describe a linear time algorithm that for a given (abstract) planar graph G and a cycle C in G tests whether C is a facial cycle of G. (You may use that planarity testing can be done in linear time.)*

In fact, we will look at a slightly different class of cycles, namely those that bound a face in *every* plane embedding of G. The lemma below provides a complete characterization of those cycles. In order to state it, let us introduce a bit more terminology. A *chord* of a cycle C in a graph G is an edge that connects two vertices of C but is not an edge of C. A cycle C in a graph G is an *induced cycle*, if $C = G[V(C)]$, that is, C does not have any chord in G.

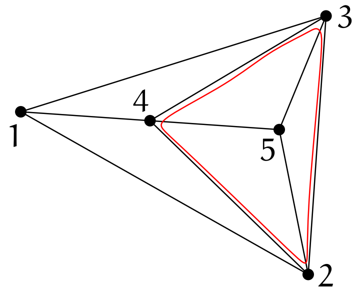


Figure 2.13: *The cycle* $(1, 2, 3)$ *is facial and we can show that* $(2, 3, 4)$ *is not.*

Lemma 2.24. *Let* C *be a cycle in a planar graph* G *such that* $G \neq C$ *and* G *is not* C *plus a single chord of* C*. Then* C *bounds a face in every plane embedding of* G *if and only if* C *is an induced cycle and it is not separating (i.e.,* $G \setminus C$ *is connected).*

Proof. "$\Leftarrow$": Consider any plane embedding Γ of G. As $G \setminus C$ is connected, by the Jordan Curve Theorem it is contained either in the interior of C or in the exterior of C in Γ. In either case, the other component of the plane is bounded by C, because there are no edges among the vertices of C.

"$\Rightarrow$": Using contraposition, suppose that C is not induced or $G \setminus C$ is disconnected. We have to show that there exists a plane embedding of G in which C does not bound a face.

If C is not induced, then there is a chord c of C in G. As $G \neq C \cup c$, either G has a vertex v that is not in C or G contains another chord $d \neq c$ of C. In either case, consider any plane embedding Γ of G in which C bounds a face. (If such an embedding does not exist, there is nothing to show.) We can modify Γ by drawing the chord c in the face bounded by C to obtain an embedding Γ' of G in which C does not bound a face: one of the two regions bounded by C according to the Jordan Curve Theorem contains c and the other contains either the vertex v or the other chord d.

If $G \setminus C$ contains two components A and B, then consider a plane embedding Γ of G. If C is not a face in Γ, there is nothing to show. Hence suppose that C is a face of Γ (Figure 2.14a). From Γ we obtain induced plane embeddings Γ_A of $G \setminus B \supseteq A \cup C$ and Γ_B of $B \cup C$. Using Theorem 2.2 we may suppose that C bounds the outer face in Γ_A and it does not bound the outer face in Γ_B. Then we can glue both embeddings at C, that is, extend Γ_B to an embedding of G by adding Γ_A within the face bounded by C (Figure 2.14b). The resulting embedding is a plane drawing of G in which C does not bound a face.

Finally, consider the case that $G \setminus C = \emptyset$ (which is not a connected graph according to our definition). As we considered above the case that C is not an induced cycle, the only remaining case is $G = C$, which is excluded explicitly. □

For both special cases for G that are excluded in Lemma 2.24 it is easy to see that all cycles in G bound a face in every plane embedding. This completes the characterization. Also observe that in these special cases G is not 3-connected.

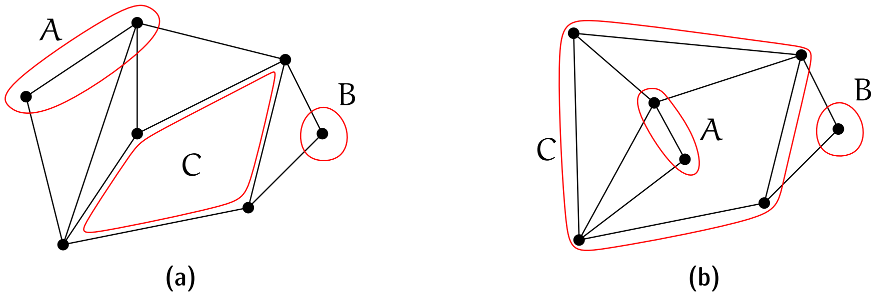


Figure 2.14: *Construct a plane embedding of G in which C does not bound a face.*

Corollary 2.25. *A cycle C of a 3-connected planar graph G bounds a face in every plane embedding of G if and only if C is an induced cycle and it is not separating.* □

The following theorem tells us that for a wide range of graphs we have little choice as far as a plane embedding is concerned, at least from a combinatorial point of view. Geometrically, there is still a lot of freedom, though.

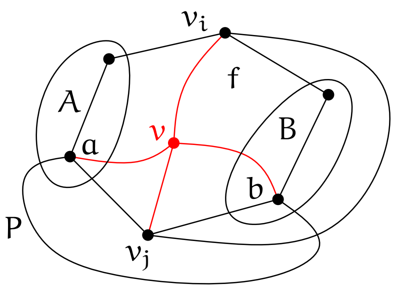
Theorem 2.26 (Whitney [36]). *A 3-connected planar graph has a unique combinatorial plane embedding (up to equivalence).*

Proof. Let G be a 3-connected planar graph and suppose there exist two embeddings Φ_1 and Φ_2 of G that are not equivalent. That is, there is a cycle $C = (v_1, \ldots, v_k)$, $k \geqslant 3$, in G that bounds a face in, say, Φ_1 but C does not bound a face in Φ_2. By Corollary 2.25 such a cycle has a chord or it is separating. We consider both options.

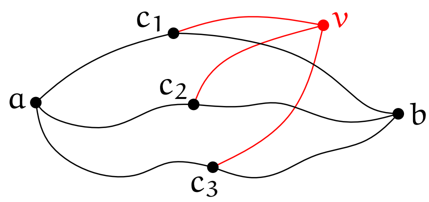
Case 1: C has a chord $\{v_i, v_j\}$, with $j \geqslant i+2$. Denote $A = \{v_x : i < x < j\}$ and $B = \{v_x : x < i \vee j < x\}$ and observe that both A and B are nonempty (because $\{v_i, v_j\}$ is a chord and so v_i and v_j are not adjacent in C). Given that G is 3-connected, there is at least one path P from A to B that does not use either of v_i or v_j. Let a denote the last vertex of P that is in A, and let b denote the first vertex of P that is in B. As C bounds a face f in Φ_1, we can add a new vertex v inside the face bounded by C and connect v by four pairwise internally disjoint curves to each of v_i, v_j, a, and b. The result is a plane graph $G' \supset G$ that contains a subdivision of K_5 with branch vertices v, v_i, v_j, a, and b. By Kuratowski's Theorem (Theorem 2.10) this contradicts the planarity of G'.

Case 2: C is induced and separating. Then $G \setminus C$ contains two distinct components A and B. (We have $V(G) \neq V(C)$ and, in particular, $G \setminus C \neq \emptyset$ because C is induced and G is 3-connected.) Consider now the embedding Φ_1 in which C bounds a face, without loss of generality (Theorem 2.2) a bounded face f. Hence both A and B are embedded in the exterior of f.

Choose vertices $a \in A$ and $b \in B$ arbitrarily. As G is 3-connected, by Menger's Theorem (Theorem 1.2), there are at least three pairwise internally vertex-disjoint paths from a to b. Fix three such paths $\alpha_1, \alpha_2, \alpha_3$ and let c_i be some point of α_i that is on C, for $1 \leqslant i \leqslant 3$. Note that c_1, c_2, c_3 exist because C separates A and B, and they are pairwise distinct because $\alpha_1, \alpha_2, \alpha_3$ are pairwise internally vertex-disjoint. Therefore,



(a) Case 1.



(b) Case 2.

Figure 2.15: *Illustration of the two cases in Theorem 2.26.*

$\{a, b\}$ and $\{c_1, c_2, c_3\}$ are branch vertices of a $K_{2,3}$ subdivision in G. We can add a new vertex v inside the face bounded by C and connect v by three pairwise internally disjoint curves to each of c_1, c_2, and c_3. The result is a plane graph $G' \supset G$ that contains a $K_{3,3}$ subdivision. By Kuratowski's Theorem (Theorem 2.10) this contradicts the planarity of G'.

In both cases we arrived at a contradiction and so there does not exist such a cycle C. Thus Φ_1 and Φ_2 are equivalent. □

Whitney's Theorem does not provide a characterization of unique embeddability in general because there are both biconnected graphs that have a unique plane embedding (such as cycles) and biconnected graphs that admit several nonequivalent plane embeddings (for instance, a triangulated pentagon).

Exercise 2.27. *Describe a family of biconnected planar graphs with exponentially many (combinatorial) plane embeddings. That is, show that there exists a constant* $c \in \mathbb{R}$ *such that for every* $n \in \mathbb{N}$ *there exists a biconnected planar graph on* n *vertices that has at least* c^n *pairwise distinct (combinatorial) plane embeddings.*

2.4 Triangulating a Plane Graph

We like to study worst case scenarios not so much to dwell on "how bad things could get" but rather—phrased positively—because worst case examples provide universal bounds of the form "things are always at least this good". Most questions related to embeddings get harder the more edges the graph has because every additional edge needs to avoid potential crossings with other edges. Therefore, let us study the class of maximal planar graphs. A graph is *maximal planar* if no edge can be added so that the resulting graph is still planar. Corollary 2.5 tells us that a (maximal) planar graph on n vertices cannot have more than $3n - 6$ edges. Yet we would like to learn a bit more about how these graphs look like.

Lemma 2.28. *A maximal planar graph on* $n \geqslant 3$ *vertices is biconnected.*

Proof. Consider a maximal planar graph $G = (V, E)$. We may suppose that G is connected because adding an edge between two distinct components of a planar graph maintains planarity. Therefore, if G is not biconnected, then it has a cut-vertex v. Take a plane drawing Γ of G. As $G \setminus v$ is disconnected, removal of v also splits $N_G(v)$ into at least two components. Therefore, there are two vertices $a, b \in N_G(v)$ that are adjacent in the circular order of vertices around v in Γ and are in different components of $G \setminus v$. In particular, $\{a, b\} \notin E$ and we can add this edge to G (routing it very close to the path (a, v, b) in Γ) without violating planarity. This is in contradiction to G being maximal planar and so G is biconnected. □

Lemma 2.29. *In a maximal planar graph on $n \geqslant 3$ vertices, all faces are topological triangles, that is, every face is bounded by exactly three edges.*

Proof. Consider a maximal planar graph $G = (V, E)$ and a plane drawing Γ of G. By Lemma 2.28 we know that G is biconnected and so by Lemma 2.20 every face of Γ is bounded by a cycle. Suppose that there is a face f in Γ that is bounded by a cycle $v_0, \dots, v_{k-1}$ of $k \geqslant 4$ vertices. We claim that at least one of the edges $\{v_0, v_2\}$ or $\{v_1, v_3\}$ is not present in G.

Suppose to the contrary that $\{\{v_0, v_2\}, \{v_1, v_3\}\} \subseteq E$. Then we can add a new vertex v' in the interior of f and connect v' inside f to all of v_0, v_1, v_2, v_3 by an edge (curve) without introducing a crossing. In other words, given that G is planar, also the graph $G' = (V \cup \{v'\}, E \cup \{\{v_i, v'\} : i \in \{0, 1, 2, 3\}\})$ is planar. However, v_0, v_1, v_2, v_3, v' are branch vertices of a K_5 subdivision in G': v' is connected to all other vertices within f, along the boundary ∂f of f each vertex v_i is connected to both $v_{(i-1) \bmod 4}$ and $v_{(i+1) \bmod 4}$ and the missing two connections are provided by the edges $\{v_0, v_2\}$ and $\{v_1, v_3\}$ (Figure 2.16a). By Kuratowski's Theorem this is in contradiction to G' being planar. Therefore, one of the edges $\{v_0, v_2\}$ or $\{v_1, v_3\}$ is not present in G, as claimed.

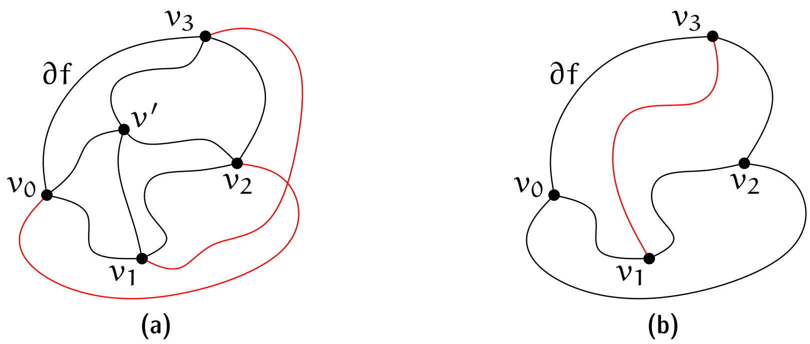


Figure 2.16: *Every face of a maximal planar graph is a topological triangle.*

So suppose without loss of generality that $\{v_1, v_3\} \notin E$. But then we can add this edge (curve) within f to Γ without introducing a crossing (Figure 2.16b). It follows that the edge $\{v_1, v_3\}$ can be added to G without sacrificing planarity, which is in contradiction to G being maximal planar. Therefore, there is no such face f bounded by four or more vertices. □

Exercise 2.30. *(a) A* minimal nonplanar *graph is a graph* G *that contains an edge* e *such that* $G \setminus e$ *is planar. Prove or disprove: Every minimal nonplanar graph contain an edge* e *such that* $G \setminus e$ *is maximal planar.*
(b) A maximal-plus-one planar *graph is a graph* G *that contains an edge* e *such that* $G \setminus e$ *is maximal planar. Prove or disprove: Every maximal-plus-one planar graph be drawn with at most one crossing (edge-pair).*

Many questions for graphs are formulated for connected graphs only because it is easy to add edges to a disconnected graph to make it connected. For similar reasons many questions about planar embeddings are formulated for maximal planar graphs only because it is easy to add edges to a planar graph so as to make it maximal planar. Well, this last statement is not entirely obvious. Let us look at it in more detail.

An *augmentation* of a given planar graph $G = (V, E)$ to a maximal planar graph $G' = (V, E')$ with $E' \supseteq E$ is also called a *topological triangulation*. The proof of Lemma 2.29 already contains the basic idea for an algorithm to topologically triangulate a plane graph.

Theorem 2.31. *For a given connected plane graph* $G = (V, E)$ *on* n *vertices one can compute in* $O(n)$ *time and space a maximal plane graph* $G' = (V, E')$ *with* $E \subseteq E'$.

Proof. Suppose, for instance, that G is represented as a DCEL[2], from which one can easily extract the face boundaries. If some vertex v appears several times along the boundary of a single face, it is a cut-vertex. We fix this by adding an edge between the two neighbors of all but the first occurrence of v. This can easily be done in linear time by maintaining a counter for each vertex on the face boundary. The total number of edges and vertices along the boundary of all faces is proportional to the number of edges in G, which by Corollary 2.5 is linear. Hence we may suppose that all faces of G are bounded by a cycle.

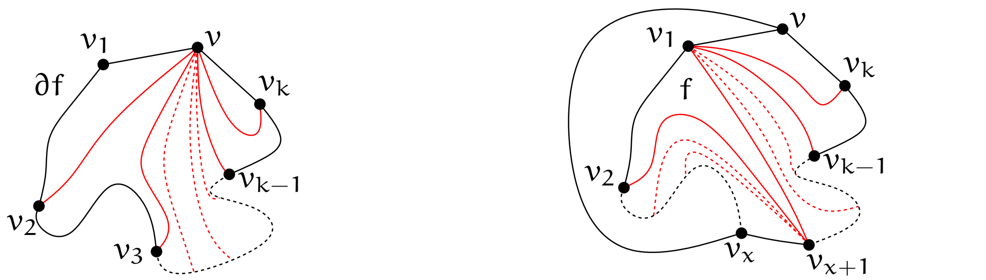
For every face f that is bounded by more than three vertices, select a vertex v_f on its boundary and store with every vertex all faces that select it. Then process every vertex v as follows: First mark all neighbors of v in G. Then process all faces that selected v. For each such face f with $v_f = v$ iterate over the boundary $\partial f = (v, v_1, \ldots, v_k)$, where $k \geqslant 3$, of f to test whether there is any marked vertex other than the two neighbors v_1 and v_k of v along ∂f.

If there is no such vertex, we can safely triangulate f using a star from v, that is, by adding the edges $\{v, v_i\}$, for $i \in \{2, \ldots, k-1\}$, and marking the new neighbors of v accordingly (Figure 2.17a).

Otherwise, let v_x be the first marked vertex in the sequence $v_2, \ldots, v_{k-1}$. The edge $\{v, v_x\}$ that is embedded as a curve in the exterior of f prevents any vertex from

[2] If you wonder how the—possibly complicated—curves that correspond to edges are represented: they do not need to be, since here we need a representation of the combinatorial embedding only.

$v_1, \dots, v_{x-1}$ from being connected by an edge in G to any vertex from $v_{x+1}, \dots, v_k$. (This is exactly the argument that we made in the proof of Lemma 2.29 above for the edges $\{v_0, v_2\}$ and $\{v_1, v_3\}$, see Figure 2.16a.) In particular, we can safely triangulate f using a bi-star from v_1 and v_{x+1}, that is, by adding the edges $\{v_1, v_i\}$, for $i \in \{x+1, \dots, k\}$, and $\{v_j, v_{x+1}\}$, for $j \in \{2, \dots, x-1\}$ (Figure 2.17b).



(a) Case 1: v does not have any neighbor on ∂f other than v_1 and v_k.

(b) Case 2: v has a neighbor v_x on ∂f other than v_1 and v_k.

Figure 2.17: *Topologically triangulating a plane graph.*

Finally, conclude the processing of v by removing all marks on its neighbors.

Regarding the runtime bound, note that every face is traversed a constant number of times. In this way, each edge is touched a constant number of times, which by Corollary 2.5 uses linear time overall. Similarly, marking the neighbors of a chosen vertex is done at most twice (mark und unmark) per vertex. Therefore, the overall time needed can be bounded by $\sum_{v \in V} \deg_G(v) = 2|E| = O(n)$, using the Handshaking Lemma and Corollary 2.5. □

Theorem 2.32. *A maximal planar graph on $n \geqslant 4$ vertices is 3-connected.*

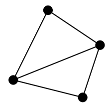
Exercise 2.33. *Prove Theorem 2.32.*

Using any of the standard planarity testing algorithms we can obtain a combinatorial embedding of a planar graph in linear time. Together with Theorem 2.31 this yields the following

Corollary 2.34. *For a given planar graph $G = (V, E)$ on n vertices one can compute in $O(n)$ time and space a maximal planar graph $G' = (V, E')$ with $E \subseteq E'$.* □

The results discussed in this section can serve as a tool to fix the combinatorial embedding for a given graph G: augment G using Theorem 2.31 to a maximal planar graph G′, whose combinatorial embedding is unique by Theorem 2.26.

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 introducing a 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 references 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-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 the parameter 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].

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] $O(n \log n)$ or—using more elaborate tools—$O(n \log^* n)$ time

[4] The result was announced in 2015 and published in 2020.

[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.