Triangulating Planar Graphs While Minimizing the Maximum Degree

Goos Kant, Hans L. Bodlaender

RUU-CS-92-07
February 1992

Utrecht University

Department of Computer Science

Padualaan 14, P.O. Box 80.089,
3508 TB Utrecht, The Netherlands,
Tel. : ... + 31 - 30 - 531454

Triangulating Planar Graphs While Minimizing the Maximum Degree

Goos Kant, Hans L. Bodlaender

Technical Report RUU-CS-92-07
February 1992

Department of Computer Science
Utrecht University
P.O.Box 80.089
3508 TB Utrecht
The Netherlands

ISSN: 0924-3275

Triangulating Planar Graphs While Minimizing the Maximum Degree*

Goos Kant Hans L. Bodlaender
Dept. of Computer Science, Utrecht University
P.O. Box 80.089, 3508 TB Utrecht, the Netherlands

Abstract

In this paper we study the problem of triangulating a planar graph G while minimizing the maximum degree $\Delta(G')$ of the resulting triangulated planar graph G'. It is shown that this problem is NP-complete. Worst-case lower bounds for $\Delta(G')$ with respect to $\Delta(G)$ are given. We describe a linear algorithm to triangulate planar graphs, for which the maximum degree of the triangulated graph is only a constant larger than the lower bounds. Finally we show that triangulating one face while minimizing the maximum degree can be achieved in polynomial time. We use this algorithm to obtain a polynomial exact algorithm to triangulate the interior faces of an outerplanar graph while minimizing the maximum degree.

1 Introduction

Planarity has been deeply investigated both in combinatorics and in graph algorithms research. Concerning undirected graphs, there are elegant characterizations of the graphs that have a planar representation and efficient algorithms for testing planarity (see, for instance, [1, 11, 7]). Also several algorithms are known to draw a planar graph in the plane with straight lines, leading to a so-called Fáry embedding. Elegant and simple algorithms for this problem are described in [6, 9, 19, 21], which all assume that the planar graph is triangulated. In general, this is no problem, since every planar graph can be triangulated in linear time and space, by a modification of the algorithm in [19]. However, this may increase the degree of any vertex by $O(n)$, even if the maximum degree of the given graph is bounded by a constant. As a concequence, the resulting picture may be significantly less readable, since a high degree implies small angles between adjacent edges.

*This work was supported by the ESPRIT Basic Research Actions of the EC under contract No. 3075 (project ALCOM).

The problem of drawing planar graphs readable in the plane gains a lot of interest for several other subclasses of planar graphs as well. Especially for biconnected and triconnected planar graphs some interesting and satisfying characterizations are known, leading to presentable drawing algorithms [3, 22, 23]. However, these algorithms, if they can be generalized, will not always give compact and structured pictures for a more general subclass of planar graphs.

Recently, an investigation of augmenting a planar graph minimally by adding edges to admit these constraints has been presented in [16]. In this paper, Kant & Bodlaender present polynomial algorithms to make a planar graph biconnected or triconnected and still planar, working within 3/2 and 5/4 times optimal, respectively. Similar optimal results with respect to outerplanar graphs are described in [14] and augmentation algorithms, without preserving planarity, are described in [4, 12, 13, 20].

In this paper we consider the problem of triangulating a planar graph while minimizing the maximum degree. We show that this problem is NP-complete in general. Let $\Delta(G)$ be the maximum degree of a planar graph G and let $\Delta(G')$ be the maximum degree of some triangulation G' of G, then our goal is that $\Delta(G') \leq c_1\Delta(G) + c_2$, with c_1 (and c_2) as small as possible. We present worst-case lower bounds for c_1 and c_2, when G is a biconnected or triconnected planar graph. The added edges $\in G - G'$ are also called augmenting.

We present a linear algorithm to triangulate a planar graph G, such that the maximum degree in the resulting triangulated graph G' is only an additive constant larger than the worst-case lower bounds. To achieve this goal, a linear algorithm is presented to biconnect a planar graph while preserving planarity, increasing the degree of any vertex with at most 2. We also define a new ordering for the vertices and faces of a triconnected planar graph. This ordering is based on the canonical ordering of [6], and is interesting in its own sense. If only one face needs to be triangulated, then a simple dynamic programming approach can be applied to find an optimal solution. Using this technique, we present a polynomial exact algorithm to triangulate the interior faces of an outerplanar graph while minimizing the maximum degree. In case the outerplanar graph is biconnected, this leads to a maximal outerplanar graph.

This paper is organized as follows. In section 2 we prove that triangulating planar graphs while minimizing the maximum degree is NP-complete for biconnected planar graphs. In section 3 we give lower bounds for the maximum degree of the triangulated graphs. In section 4 we present a linear algorithm for triangulating planar graphs, working only an additive constant from the lower bounds. In section 5 we present an algorithm for triangulating one face. In section 6 we present an algorithm for triangulating outerplanar graphs. Section 7 contains some concluding remarks and some open problems.

2 NP-completeness

Triangulated planar graphs have several properties in drawing algorithms (see [19, 6, 14, 9]). A triangulated planar graph has $3n - 6$ edges and adding any edge to it destroys the planarity. Every face is a triangle. From an aesthetic point of view for the drawing algorithms, we want to triangulate a graph G (with n vertices and m edges) while minimizing its maximum degree. This is a hard problem, as stated in the following main result of this section.

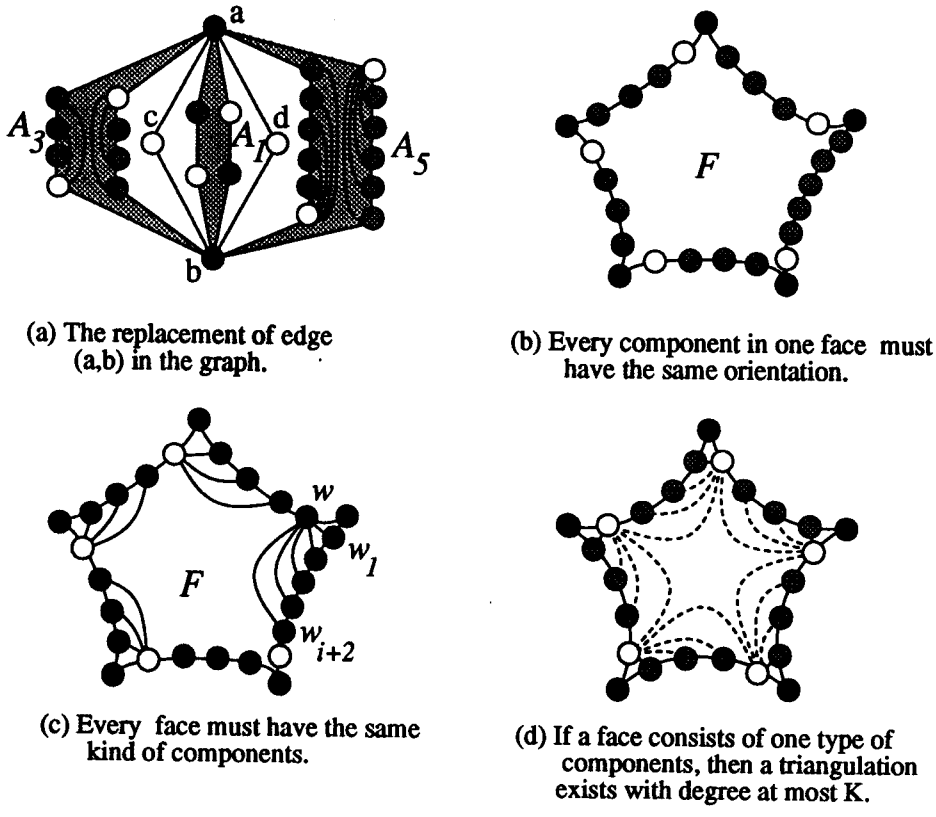
Theorem 2.1 *Deciding whether a biconnected planar graph can be triangulated such that the maximum degree is $\leq K$ is NP-complete.*

Proof: The problem is in NP: guess $3n - 6 - m$ additional edges, and test in polynomial time whether G is planar and has maximum degree $\leq K$.

To prove the NP-hardness we use a reduction from the 3-coloring problem for triconnected planar graphs. It is well-known that deciding whether a planar graph can be colored with three colors such that every pair of two neighbors have different colors is NP-complete [8]. It is easy to see that the proof in [8] can be modified, such that the NP-completeness of the 3-coloring problem also follows for triconnected planar graphs. We omit the details here. Let a triconnected planar graph H be given, and let $K \geq 12\Delta(H)$ be defined. H has an unique embedding and let G be the dual graph of H: every vertex of G correspond with a face of H and there is an edge between two vertices in G, if and only if the corresponding faces in H share a common edge. G is triconnected and planar as well. We change graph G into a graph G' as follows: we replace every edge $(a, b) \in G$ by three components A_1, A_3, A_5 and two vertices c, d with edges to a and b as shown in figure 1(a). A_i is a *triangulated* component between a and b. The outerface of A_i consists at each side of a and b of i consecutive vertices $v_1, \ldots, v_i$ and $w_1, \ldots, w_i$ of degree $K - 1$ and one vertex of degree $K - i - 2$. These vertices v, w with $deg(v) = deg(w) = K - i - 2$ are adjacent to a and b, respectively, and to $v_1, \ldots, v_i$ and $w_1, \ldots, w_i$, respectively (see figure 1(a)).

We add vertices inside some A_i-components with edges to a and b in such way that the degree for all vertices a and $b \in G$ is K in G' and the degree of the other vertices on the outerface of the triangulated A_i-components does not change. We call the vertices $v \in G'$ with $deg(v) < K - 1$ *white*, with $deg(v) = K - 1$ *grey* and with $deg(v) = K$ *black*, and they are drawn accordingly in figure 1. Notice that the vertices of G are black in G' and the other relevant vertices in G' are grey or white. All vertices in an A_i-component have degree $\leq K$; all faces inside an A_i-component are triangulated. (It is not hard to find the precise constructions for A_1, A_3, A_5; although a little tedious. We omit the precise construction.)

Suppose G has a triangulation with maximum degree $\leq K$. Fix such a triangulation, and a planar embedding of the triangulated graph.



(a) The replacement of edge (a,b) in the graph.

(b) Every component in one face must have the same orientation.

(c) Every face must have the same kind of components.

(d) If a face consists of one type of components, then a triangulation exists with degree at most K.

Figure 1: Figures for the NP-completeness proof.

Lemma 2.2 *Between every two components A_i, A_j with common vertices a, b there must be a vertex c or d.*

Proof: Suppose not, i.e., there are two components A_i, A_j, assume $j > i$, adjacent to each other between two vertices a and b in G'. Let F be the face between these components. If there are two grey vertices adjacent to a or b in F, then one of these grey vertices must get two extra edges by the triangulations, hence will get degree $> K$, which is not allowed. So assume that adjacent to a and b there is a white and a grey vertex. If we want to triangulate F such that every vertex has degree $\leq K$, then each consecutive sequence of grey vertices must get incident edges to a common white vertex. Let v, w be the white vertices and $v_1, \ldots, v_i$, $w_1, \ldots, w_j$ be the grey vertices of F_1, then we must assign $i + 2$ edges between v and $w_1, \ldots, w_{i+2}$ and i edges between w and $v_1, \ldots, v_i$. After this assignment, v and w_{i+2} are now both black, but since $j \geq i + 2$, F is not completely triangulated yet. But v and w_{i+2} are now neighbors in F, and one of them must get an extra incident edge. Contradiction with the assumption that we can triangulate F such that the maximum degree is $\leq K$. Thus between every two components A_i, A_j of vertices a

and b, there must be a vertex c or d. □

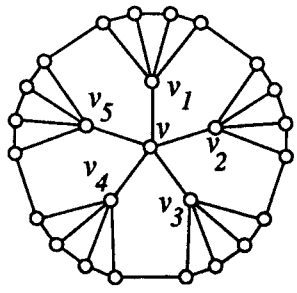
This means that for every edge $(a, b) \in G$, belonging to faces F_1, F_2, we have to assign one A_i-component of a, b to F_1 and one A_j-component of a, b to F_2, with $i \neq j$. Assigning means that this component lies at the outside at the corresponding face. Triangulating these inside faces can simply be done by adding edges from vertex c or d to all other vertices of these faces.

Lemma 2.3 *To every face $F \in G'$, only one type of component A_i can be assigned to it.*

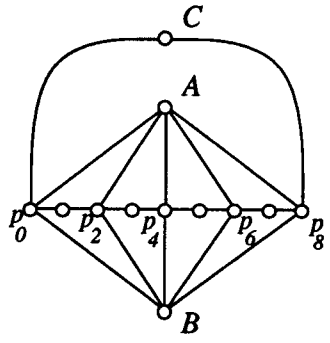
Proof: Suppose we can triangulate a face F, where different components $A_i, A_j, j > i$ are assigned to it, such that the maximum degree is still $\leq K$. First we notice that every black vertex $v \in F$ may not get an augmenting edge, hence there must come an edge between its two neighbors $v_1, v_2 \in F$. Since for any triangulation v_1 or v_2 must get in total at least two augmenting edges, one of them must be white. This means that every component in F must have the same orientation, i.e., when walking around F we visit alternatingly a black vertex, a white vertex and a sequence of grey vertices (see figure 1(b)). Notice that after adding (v_1, v_2) for every black vertex v in F, the size of F decreases, but since v_1 or v_2, say v_1, was grey, v_1 is black now, and we repeat our argument for all black vertices v_1. Inspect the two adjacent components $A_i, A_j, j > i$, of a black vertex v, where after assigning $i + 2$ edges from the white vertex w of A_i to the grey vertices $w_1, \ldots, w_{i+2}$ of A_j, w and $w_1, \ldots, w_{i+2}$ are all black now. But similar as in Lemma 2.2, since $j \geq i + 2$, F is not completely triangulated yet, thus w or w_{i+2} must receive at least one extra edge (see figure 1(c)). Contradiction with the assumption that we could triangulate F such that the maximum degree is $\leq K$. □

If only one type of components A_i is assigned to a face $F \in G'$ then we can triangulate F as follows: from every white vertex we add i edges to the grey vertices of the next component in the circular order of F. In the reduced face F we assign edges between every two consecutive white vertices. This triangulates F completely and the degree of every vertex $v \in F$ becomes K. An example is given in figure 1(d).

From the two lemmas and the construction in figure 1(d) it follows that we can triangulate G' with maximum degree $\leq K$ if and only if we can assign to every face $F \in G'$ only one type of components A_i, $i = 1, 3, 5$, i.e., if and only if we can assign one number $i, i = 1, 3, 5$, to every face in G such that every two faces, sharing a common edge in G, have a different number, i.e, if and only if there exists a coloring of H with three colors such that every pair of two neighbors v, w, have a different color. As G and G' can be constructed in time polynomial in K, n and m, there is a polynomial time transformation from the NP-complete problem of 3-coloring triconnected planar graphs to the problem of triangulating a planar graph while



(a) Example of theorem 2.1 with $\Delta = 5$.



(b) Example of theorem 2.2 with $\Delta = 5$.

Figure 2: Examples of the lower bounds for the maximum degree.

minimizing the maximum degree, hence the latter is NP-complete. □

3 Lower bounds for $\Delta(G')$

Regarding the problem of triangulating G into G' such that $\Delta(G') \leq c_1\Delta(G) + c_2$, we present here lower bounds for c_1 and c_2, when G is a biconnected or triconnected planar graph.

Theorem 3.1 *For every $\Delta = \Delta(G) \geq 5$, there exist triconnected planar graphs G such that for every triangulation G' of G: $\Delta(G') \geq \Delta(G) + 3$.*

Proof: Construct a triconnected planar graph G as follows. $V = \{v, v_i, v_{ij} | 1 \leq i \leq \Delta, 1 \leq j \leq \Delta - 1\}$, thus $|V| = \Delta^2 + 1$. Connect v to the Δ vertices $v_1, \ldots, v_\Delta$. Connect v_i to $\Delta - 1$ vertices v_{ij}, $1 \leq j \leq \Delta - 1$. Connect v_{ij} to $v_{i(j+1)}$ if $j < \Delta - 1$ and to $v_{(i+1)1}$ otherwise (see figure 2(a)). Suppose we can triangulate G such that $\Delta(G') \leq \Delta(G) + 2$, then v must get at most two augmenting edges. Let $F_1, \ldots, F_\Delta$ be the faces adjacent to v. F_i consists of 5 vertices, thus every triangulation of F_i consists of two augmenting edges, incident to one vertex $w \in F_i$, which we call *marked*, to its non-neighbors of F_i. Hence except for at most two neighbors v_i, v_j of v, all other $\Delta - 2$ neighbors of v must be marked. But marking both v_i and v_{i+1} increases $deg(v_i)$ by 3. Hence only half of the neighbors of v can be marked, which is $> \Delta - 2$ for $\Delta \geq 5$. Thus $\Delta(G') \geq \Delta + 3$, for $\Delta \geq 5$. □

Theorem 3.2 *For every $\Delta > 1$ there exists a biconnected planar graph G for which for every triangulation $G', G \subseteq G'$: $\Delta(G') \geq \lceil \frac{3}{2}\Delta(G) \rceil$.*

Proof: We can assume w.l.o.g. that $\Delta = \Delta(G) \geq 4$. (For $\Delta = 2, 3$ the theorem trivially holds by constructing a graph containing a cycle of length 5.) Construct the graph G_Δ consisting of two vertices A and B, and $2\Delta - 1$ vertices $p_0, \ldots, p_{2\Delta-2}$. There is an edge (p_i, p_{i+1}) for $0 \leq i < 2\Delta - 2$. There are edges (A, p_i) and (B, p_i), for i even, $0 \leq i \leq 2\Delta - 2$. For Δ odd (implying $\Delta \geq 5$), a separate vertex C is inserted with edges (p_0, C) and $(C, p_{2\Delta-2})$. See figure 2(b).

In every face F_i with vertices p_i, p_{i+1}, p_{i+2}, A and in every face F_i' with vertices p_i, p_{i+1}, p_{i+2}, B, one extra edge must be added (i even). When we add an edge (p_i, p_{i+2}) in F_i, then an edge (p_{i+1}, B) must be added in F_i'. Since there are $\Delta - 1$ faces F_i, this means that the total increase of the degree of A and B is $\Delta - 1$. If Δ is odd, then an additional edge has to go from C to A or B. Hence the degree of A or B increases by at least $\lceil \frac{\Delta}{2} \rceil$. □

This theorem shows that $c_1 \geq \frac{3}{2}$ for planar graphs in general. In the following section we will present an approximation algorithm, which will match these bounds up to an additive constant.

4 An Approximation Algorithm

4.1 Introduction

Our algorithm in this section will lead to a proof of the following theorem:

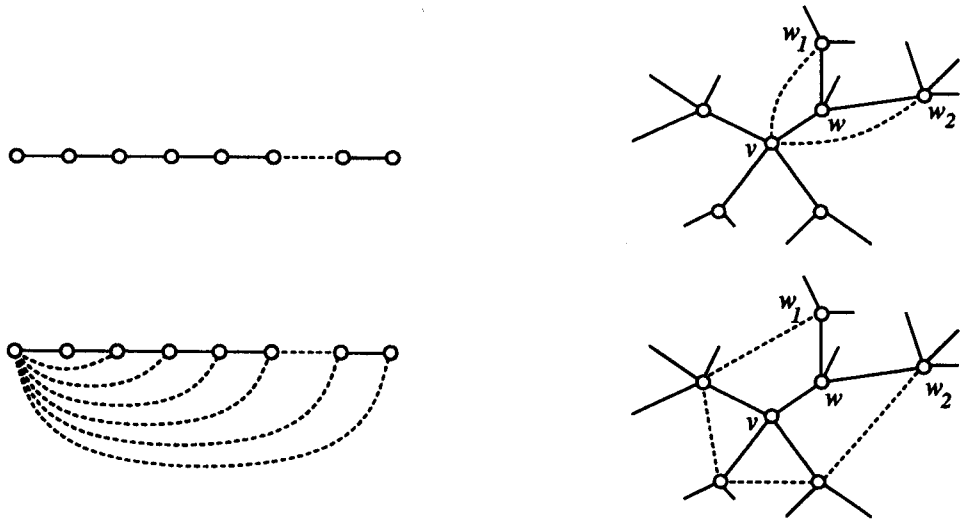
Theorem 4.1 *There is a linear algorithm to triangulate a planar graph G such that for the triangulation G' of G, $\Delta(G') \leq \lceil \frac{3}{2}\Delta(G) \rceil + 21$.*

A brief outline of the algorithm which obtains this result is as follows:

TRIANGULATE
  make the graph biconnected;
  while the entire graph is not triangulated **do**
    determine a triconnected component G' of G;
    compute the *canonical 3-ordering* for G';
    triangulate G';
    replace G' by an edge between the cutting pair;
  od

First we want to biconnect G such that the degree of the vertices increases as little as possible. A degree-increase of two is sometimes necessary: for instance consider the tree $K_{1,3}$. Here we show how to make G biconnected and planar such that every degree increases indeed by at most two.

The problem of augmenting G such that it is biconnected and planar by adding $\leq K$ edges is NP-complete [16], but there is a linear algorithm to make G biconnected by adding edges while preserving planarity [19]. The algorithm of [19] can



(a) One vertex gets n-2 extra edges.

(b) Every vertex gets at most 2 extra edges.

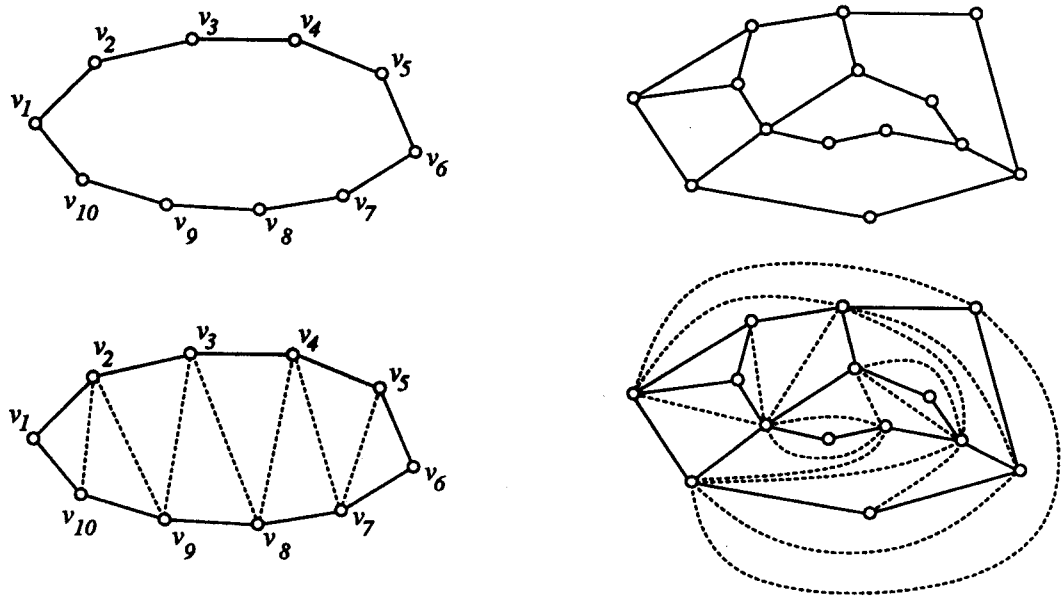
Figure 3: Making different graphs biconnected.

be described as follows: if for any pair of consecutive neighbors u, w of v, u, w belong to different biconnected components (also called *blocks*), then the edge (u, w) is added. Unfortunately, this may increase the degree of a single node by $O(n)$, as shown in figure 3(a). Therefore we modificate the algorithm of [19] by inspecting the vertices in depth-first order. Secondly, we test during the algorithm whether an added edge can be removed without destroying biconnectivity. The algorithm can now be described more formally as follows:

BICONNECT
    determine an arbitrary embedding in the plane of G;
    number the cutvertices v_i of G in depth-first order;
    for every cutvertex v_i (in increasing v_i-number) **do**
        for every two consecutive neighbors u, w of v_i in G **do**
            if u and w belong to different blocks **then**
                add an edge (u, w) to G
                if (v_i, u) or (v_i, w) was added to G **then** remove this edge;
        rof
    rof

Lemma 4.2 *Algorithm* BICONNECT *gives a biconnected planar graph.*

Proof: For every cutvertex v_i we add edges between the adjacent neighbors u, w of v_i, if they belong to different blocks, hence after this augmentation all neighbors



(a) Zigzagging one face.

(b) Zigzagging the entire planar graph.

Figure 4: Example of zigzagging a face and zigzagging a planar graph.

of v_i belong to one common block. Thus v_i is not a cutvertex anymore. If (v_i, u) was added by BICONNECT then there was a path from v_i to u initially. But then there was initially a path from w to u. Adding (u, w) implies a cycle containing v_i, hence (u, v_i) can be removed without destroying the biconnectivity. Similar for (v_i, w). □

Lemma 4.3 *Every vertex receives at most 2 augmenting incident edges.*

Proof: Assume w.l.o.g. vertex v is the only element of a block. When visiting a neighbor w of v, v receives two incident augmenting edges, say (w_1, v) and (w_2, v). Then by the depth-first order v will be visited. If v is adjacent to at least one other block, then edges between neighbors of v are added and the edges (w_1, v) and (w_2, v) are removed. Hence v receives at most two incident augmenting edges (see figure 3(b)). □

Corollary 4.4 *There is a linear algorithm to augment a planar graph such that it is biconnected and planar and increases the degree of every vertex by at most 2.*

Hence we may now assume that G is biconnected. The main method of the triangulation is the following "zigzag"-method (see also figure 4):

ZIGZAG(F, v_1, v_p);
(* F is a face of p vertices, numbered $v_1, \ldots, v_p$ around the ring; *)
add edges $(v_p, v_2), (v_2, v_{p-1}), (v_{p-1}, v_3), (v_3, v_{p-2}), \ldots, (v_{\lfloor \frac{p}{2} \rfloor}, v_{\lfloor \frac{p}{2} \rfloor + 2})$;

By this method the degree of v_1 and $v_{\lfloor \frac{p}{2} \rfloor + 1}$ does not increase, the degree of v_p and $v_{\lfloor \frac{p}{2} \rfloor}$ (p even) or $v_{\lfloor \frac{p}{2} \rfloor + 2}$ (p odd) increases by 1, all other degrees increase by 2 (see figure 4(a)).

A simple technique for triangulation is to apply ZIGZAG to all faces. However, since a vertex v belongs to $deg(v)$ faces and $deg(v)$ can increase by two in every face, this may lead to a maximum degree of $3\Delta(G)$. Moreover, this algorithm may imply multiple edges, which are not allowed (see figure 4(b)).

To apply a more subtle technique, we recall the *canonical ordering*, as introduced by De Fraysseix et al. [6]:

> It is possible to order the vertices of a triangulated planar graph G in a sequence $v_1, \ldots, v_n$ such that for $k = 3, 4, \ldots, n-1$:
>
> - the subgraph G_k of G induced by $v_1, \ldots, v_k$ is biconnected and the boundary of its exterior face is a cycle C_k containing the edge (v_1, v_2).
> - v_{k+1} is in the exterior face of G_{k+1} and has at least two neighbors in G_k, which form a consecutive sequence on the path $C_k - (v_1, v_2)$.

However, in our case the planar graph is not triangulated. We modify this ordering such that it holds for triconnected planar graphs. This ordering is then applied on the triconnected components of our biconnected planar graph G. The following simple observation of [6] will be essential for our purpose:

Lemma 4.5 *Let G be a simple planar graph embedded in the plane and $u = u_1, u_2, \ldots, u_j = v$ be a cycle of G. Then there exists a vertex w' (w'') on the cycle different from u and v and not adjacent to any inside chord (outside chord).*

Theorem 4.6 *It is possible to order the vertices of a triconnected planar graph G in a sequence $v_1, \ldots, v_n$ such that in every step $k, k \geq 2$:*

1. *The subgraph G_k of G induced by the vertices $v_1, \ldots, v_k$ is biconnected and the boundary of its exterior face is a cycle C_k.*
2. *either v_{k+1} is in the exterior face of G_{k+1} and has at least two neighbors in G_k, which are on C_k*
3. *or there exists an $l \geq 2$ such that $v_{k+1}, \ldots, v_{k+l}$ is a path in the exterior face of G_{k+l} and has exactly two neighbors in G_k, which are on C_k.*

Proof: The vertices $v_n, v_{n-1}, \ldots, v_3$ will be defined by reverse induction. Let v_n be an arbitrary vertex of the outerface not adjacent to v_1, v_2 and let G_{n-1} denote the subgraph of G after deleting v_n. By the triconnectivity of G, the outerface C_{n-1} of G_{n-1} is a cycle.

Let $i < n$ be fixed, and assume that v_k has already been determined for every $k > i$ such that the subgraph G_{k-1} induced by $V(G)\backslash\{v_k, v_{k+1}, \ldots, v_n\}$ satisfies conditions 1., 2. and 3. Let C_i denote the boundary of the exterior face of G_i. Notice that by this construction, if there are vertices $v \in G_i$ of degree 2, then $v \in C_i$ and we can take any maximal path P of vertices $v_l, v_{l+1}, \ldots, v_i$ of degree 2 as the next path P in the ordering. The subgraph G_{l-1} induced by $V(G)\backslash\{v_l, v_{l+1}, \ldots, v_n\}$ obviously meets the requirements.

Otherwise all vertices in G_i have degree > 2. Applying lemma 4.5 to the cycle C_i in G_i, we obtain that there is a vertex $w' \in C_i$, not adjacent to any chord of C_i. (Observe that C_i has no exterior chords.) Letting $v_i = w'$, the subgraph G_{i-1} induced by $V(G)\backslash\{v_i, v_{i+1}, \ldots, v_n\}$ meets the requirements. □

We call this ordering the canonical 3-ordering. The following approach can be used to find the canonical 3-ordering of a triconnected planar graph G in linear time. Let the faces be numbered $F_1, \ldots, F_f$ and let every edge e have pointers to the two faces, where e belongs to. Also every vertex v has pointers to the faces, where v belongs to. Let an embedding of G be given, i.e., let the edges incident to a vertex v be stored in a circular adjacency list in order we visit them when walking counterclockwise around v. Similar we store to each face F a circular list of edges, in counterclockwise order of visiting them in F. We give all vertices and faces a label, which can have the value (a), which means: not yet visited, (b), which means: visited once or (i), which means: visited more than once and the visited edges form i intervals in the circular edge-list of this vertex or face. Every face also contains counters i_v and i_e, indicating the number of visited vertices and edges respectively, belonging to this face. These labels are updated after we choose vertex v_{k+1} or path $v_{k+1}, \ldots, v_{k+l}$ (implying a face F_{k+1}). When choosing v_{k+1}, we visit each neighbor v of v_{k+1} along the edge connecting them. If v has label (a), then label (b) replaces label (a). If v has label (b) and the edge (v_{k+1}, v) is adjacent to a visited edge in the edge-list of v, label (1) replaces label (b) and if not, label (2) replaces label (b). Finally, if v has label (j) and the left and right neighbors of the edge (v_{k+1}, v) have already been visited then the label $(j-1)$ replaces label (j). If none of these edges have been visited then label $(j+1)$ replaces label (j), otherwise label (j) is not changed. It is clear that label (j) on v means that the edges already been visited and incident to v are composed of j intervals in the edge-list of v.

For every visited edge e we update the label for the two faces, e belongs to, and we increase the i_e label by one. When we visit a vertex v of label (a), then we increase the i_v labels of all faces, where v belongs to. When we add a path $v_{k+1}, \ldots, v_{k+l}$, then for all these vertices we update the labels of their neighbors and the incident faces.

It is easy to see that if there is a vertex with label (1), then we can choose this vertex as v_{k+2}. Otherwise by theorem 4.6 there is always a face F with label (1) and with $i_v = i_e + 1$, and we can choose the consecutive sequence of not-added vertices of F as the path $v_{k+2}, \ldots, v_{k+l}$ in the canonical 3-ordering.

Using this approach we visit every edge (u, w) exactly two times (via u and via w). Every face will be visited once via every edge and once via every vertex of it. Every triconnected planar graph has a linear number of edges and faces, completing the proof of the following result:

Theorem 4.7 *The canonical 3-ordering of a triconnected planar graph can be computed in linear time and space.*

We use the canonical 3-ordering for triangulating the interior faces of each triconnected component of G. If deleting two vertices a, b of G disconnects G into one connect component V' and some components $G' - V'$, then we call V' a *2-subgraph*, and a, b is called the *cutting pair* of V'. If V' contains no other 2-subgraphs then V' is a triconnected component. Otherwise, if we replace all 2-subgraphs of V' by an edge between the cutting pair and eliminate the multiple edges, then V' is also a triconnected component. Notice that adding an edge between the cutting pair of a triconnected component V' makes V' triconnected. Hence with a small modification, we can apply the canonical 3-ordering on the 2-subgraphs of G.

Let T be a tree, where every node v' of T corresponds with a 2-subgraph V' of G. v' is an ancestor of w' in T, if $W' \subseteq V'$. The root r of T corresponds with the entire graph G. The leaves of T corresponds with the triconnected components of G. We call T the *2-subgraph-tree* (see figure 5). We start with triangulating the internal faces of the triconnected components of G and then the other 2-subgraphs V' of G, if all 2-subgraphs $W' \subseteq V'$ are already triangulated. To compute the canonical 3-ordering for V', we replace the 2-subgraphs $W' \subseteq V$ by an edge (a, b) between the cutting pair, and we *mark* this edge (a, b). When we triangulate V' and we visit edge (a, b), then we replace (a, b) by W'. We call this procedure *folding* and *unfolding*. Actually, this means that we triangulate V', if all 2-subgraphs W', for which v' is an ancestor of w', are already triangulated. We start with the leaves of T and work towards the root r of T.

We consider the two cases of the canonical 3-ordering: either a vertex v_{k+1} or a path $v_{k+1}, \ldots, v_{k+l}$ (implying a face F_{k+l}) will be added. When adding a vertex, several faces need to be triangulated, for which we use the algorithm for triangulating one face. We call in a step k the vertices $v_1, \ldots, v_k$ *old* and the vertices added in step $k+1$ *new*. Let v_a and v_b be the *leftmost* and *rightmost* old neighbor of the new added vertices in step $k+1$, then we call the old vertices $v \in C_k$ between v_a and v_b *internal*. Notice that every vertex is exactly once new and once internal, but it can be left- or rightmost more than once. Vertex v may receive in each phase (new, left- or rightmost, internal) incident augmenting edges. The upper bound for the increase of $deg(v)$ is the sum of the upper bounds for the increases in each of the

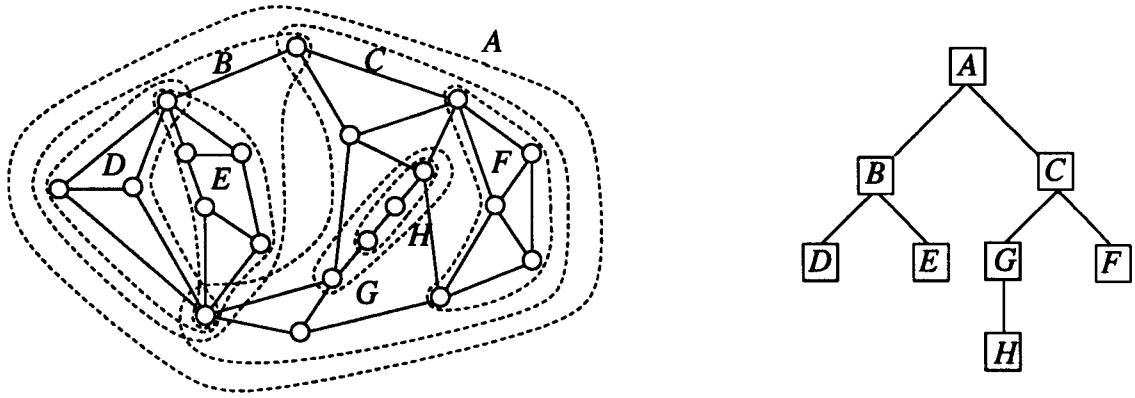


Figure 5: Example of a biconnected planar graph and the corresponding 2-subgraph-tree.

three phases. The analysis for these bounds follows by inspecting the two cases (i) adding a face and (ii) adding a vertex, in the following subsections.

4.2 Adding a face

When we add a face to G_k by adding the vertices of a path P, then we first change the marked edges (a, b) of P into the corresponding 2-subgraphs W' of G. Let F_{k+1} be the added face with all marked edges unfolded. We want to triangulate F_{k+1} such that the degree of the new and internal vertices increases by a constant, and the degree of the left- and rightmost vertex increases as little as possible.

Hereto we first assume that there is at least one internal vertex. Let $w_1, \ldots, w_r$ be the vertices of C_k such that w_1 is the leftmost and w_r the rightmost vertex, and $w_{r+1}, \ldots, w_l$ is the new added path (numbered counterclockwise around face F_{k+1}).

We add the edges (w_l, w_2) and (w_{r+1}, w_{r-1}), hence we need to triangulate the face with vertices $w_2, w_3, \ldots, w_{r-1}, w_{r+1}, \ldots, w_l$ and the degree of w_1 and w_r (the left- and rightmost vertex) does not increase. We call already existing edges (w_i, w_j) $2 \leq i, j \leq l$ *forbidden*, if $j > i + 1$. For every forbidden edge (w_i, w_j) we label w_{i-1} and w_{j-1}, if they do not have incident forbidden edges. We renumber the labeled vertices by $v'_1, \ldots, v'_p$, in order of increasing w_i-number. We add the edges (v'_j, v'_{j+1}), $(1 \leq j < p)$, thereby introducing several faces F''_j. Let F'_{k+1} be this face, containing all labeled vertices. We triangulate the faces as follows:

for every face F''_j **do** ZIGZAG(F''_j, v'_j, v'_{j+1});
ZIGZAG(F'_{k+1}, v'_1, v'_p);

In figure 6 an example of the triangulation is given.

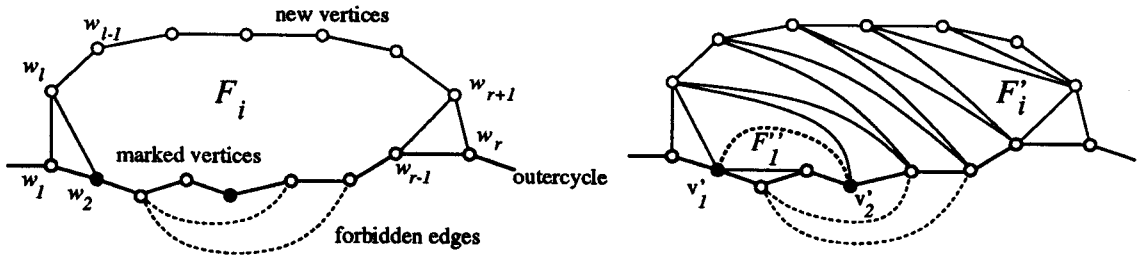


Figure 6: Example of triangulating one face.

Lemma 4.8 *The triangulation of F_{k+1} implies no multiple edges.*

Proof: Notice that if (w_i, w_j) is a forbidden edge, then there must be a vertex $w_x, i < x < j$, which is labeled. Let $v'_i = w_x$, then by adding the edges (v'_{i-1}, v'_i) and (v'_i, v'_{i+1}), there is no face F anymore with both $w_i, w_j \in F$. Hence the edge (w_i, w_j) will never be added by ZIGZAG to G. □

Lemma 4.9 *If there is an internal vertex, then the degree of the internal and new vertices increases by at most 5.*

Proof: Degrees of not labeled vertices increase by two by zigzagging F'_{k+1}. A labeled vertex v'_j gets two edges (v'_{j-1}, v'_j) and (v'_j, v'_{j+1}). Furthermore, v'_j gets two incident edges by zigzagging F'_{k+1}, one incident edge by ZIGZAG(F''_j, v'_j, v'_{j+1}), but no incident edges by ZIGZAG(F''_j, v'_{j-1}, v'_j). Hence the degree of the labeled vertices increases by at most 5; the degree of the other vertices increases by at most 2. □

The problem arises when there is no internal vertex, because then at least one outgoing edge has to go to a left- or rightmost vertex, since they are neighbors on C_k. Let F_{k+1} be the added face with left- and rightmost vertex w_1 and w_2, respectively. w_1 and w_2 cannot be labeled, thus w_1 and w_2 are both element of F'_{k+1}. Calling ZIGZAG(F'_{k+1}, w_1, w_2) increases $deg(w_2)$ by 1 and does not increase $deg(w_1)$. We introduce a counter $extra(v)$ for every vertex v, counting the augmenting incident edges of v, added when v was left- or rightmost. When there is no internal vertex, we add an edge to the left- or rightmost vertex, according to the lowest *extra*-value, by ZIGZAG(F'_{k+1}, w_2, w_1) or ZIGZAG(F'_{k+1}, w_1, w_2), respectively. Initially $extra(v) = 0$ for all $v \in V$.

Lemma 4.10 *For every four consecutive vertices v_1, v_2, v_3, v_4 on C_k, the following holds:*
If $extra(v_2) = 2$ then either $extra(v_1) = extra(v_3) = 0$ or $extra(v_1) = extra(v_4) = 0$ and $extra(v_3) = 1$. If $extra(v_2) = extra(v_3) = 1$ then $extra(v_1) = extra(v_4) = 0$.

Proof: By induction. When starting the algorithm $extra(v_1) = extra(v_2) = 0$. Assume the lemma holds for G_k. We prove that the lemma also holds for G_{k+1}, when adding at least two vertices w_1, w_2 to two consecutive old neighbors v_2 and v_3, and the lowest *extra*-value of v_2 and v_3, say v_2, increases by 1.

If $extra(v_2) = 1$, then we obtain the following *extra*-values from v_1 to v_4 on C_{k+1}: 0, 2, 0, 0, $extra(v_3)$, 0. If $extra(v_2) = 0$, then we obtain: 0, 1, 0, 0, $extra(v_3)$, 0. Hence also on C_{k+1} the lemma holds. □

Lemma 4.11 *For every pair of consecutive vertices v_2, v_3 on C_k, $extra(v_2) + extra(v_3) \leq 3$.*

Proof: Suppose not. Let k be the smallest possible integer such that for two consecutive vertices v_2, v_3 on C_k: $extra(v_2) + extra(v_3) > 3$, thus $extra(v_2) + extra(v_3) = 4$ and on C_{k-1}, $extra(v_2) + extra(v_3) = 3$. But then on C_{k-1}, $extra(v_1) = extra(v_4) = 0$, and on C_k both v_2 and v_3 have neighbors with smaller *extra*-value. The added face in C_k cannot have leftmost vertex v_2 and rightmost vertex v_3, because then vertices with *extra*-value 0 come between them. Hence $extra(v_2)$ and $extra(v_3)$ are not increased in step k. Contradiction with the assumption that in step k, $extra(v_2) + extra(v_3) > 3$. □

Corollary 4.12 *For every vertex v, $extra(v) \leq 2$.*

This implies that in every step when we add a face, the increase of $deg(v)$ is at most 5 when v is new or internal. Moreover, during all steps when adding a face, the increase of $deg(v)$, with v the left- or rightmost vertex, is totally at most 2. Hence in all phases of v, the increase of $deg(v)$ is bounded by a constant.

4.3 Adding a vertex

When we add a vertex v_{k+1} to G_k by the canonical 3-ordering, then we first replace the marked incident edges of v_{k+1} by the corresponding 2-subgraph. Assume there are t faces $F_1, \ldots, F_t$ to be triangulated, numbered from left to right. We do not want to increase the degree of the leftmost vertex $v \in F_1$ and the rightmost vertex $w \in F_t$. Therefore we add an edge between the two neighbors of v in F_1 and between the neighbors of w in F_t (only if F_1 and F_t are not triangles already). Due to these edges, $deg(v)$ and $deg(w)$ will not increase by triangulating F_1 and F_t.

Moreover, to increase $deg(v_{k+1})$ as little as possible, we add in each face F_j an edge (w_1, w_2) between the two neighbors w_1 and w_2 of v_{k+1} in face F_j (if F_j is not a triangle already). These newly added edges may imply multiple edges. In section 4.4 it is shown how this can happen and remedied. Remains now t reduced faces F_j to be triangulated. Hereto we apply the algorithm of section 4.2 to every face F_j. Doing this sequentially for $j = 1$ upto t, this will not imply multiple edges. It is easy

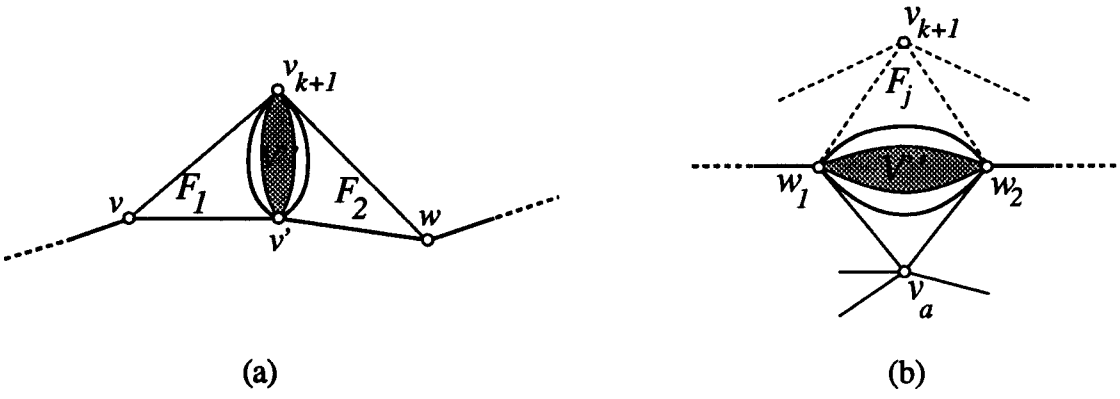


Figure 7: Reducing the faces may yield multiple edges.

to prove that an involved vertex v, belonging to two faces, F_j, F_{j+1}, can never be labeled, hence its degree will increase by at most 4. The degree of v_{k+1} will increase by at most 2 and the degree of the other vertices of F_j will increase by at most 5. Hence this increases the degree of the new vertex by at most 2 and the degree of the internal vertices by at most 5.

4.4 Eliminating multiple edges

Applying the algorithms for triangulating one face in section 4.2 does not imply multiple edges, as proved in theorem 4.8. Hence applying this algorithm sequentially to the reduced faces $F_1, \ldots, F_t$ when adding a vertex does not imply multiple edges. However, reducing the faces $F_1, \ldots, F_t$ may imply multiple edges, as can be seen as follows:

1. We first add edges between the two neighbors of v and w in F_1 and F_t, respectively. If $t = 2$, then possibly these two neighbors of v and w are equal, say v_{k+1} and v'. Hence we have a multiple edge. To eliminate one of these two edges, an edge has to go to v or w. But when this occurs several times for v or w, the increase of $deg(v)$ or $deg(w)$ is not bounded by a constant.

2. Suppose there was already an edge between the neighbors w_1 and w_2 of v_{k+1} in F_j, hence adding such an edge implies a multiple edge. To eliminate this, an edge can go from v_{k+1} in F_j. But when this occurs for several faces F_j, the increase of $deg(v_{k+1})$ is not bounded by a constant.

To eliminate all multiple edges, obtained by reducing the faces $F_1, \ldots, F_t$, we construct a bipartite planar graph H. (One set of vertices in H corresponds to a subset of the vertices in G, one set corresponds to 2-subgraphs that must have an

extra outgoing edge to remove one pair of multiple edges.) For case 1. and case 2. we do the following:

case 1. We represent the 2-subgraph V'' between v_{k+1} and v' by a vertex v'' in H. To eliminate the multiple edge, there has to go an edge from V'' to v or w. Hereto we add v and w to H (if not present already) with edges to v''.

case 2. Suppose we have the multiple edge (w_1, w_2) with the 2-subgraph V'' between these two edges. Let v_{k+1} and v_a be the other vertices, not part of V'', which have both w_1 and w_2 as neighbors (as in figure 7(b)). Then an edge has to go from v_a or v_{k+1} to V''. Hereto we represent V'' by a vertex v'' in H. Add the vertices v_a and v_{k+1} to H (if not present already) with edges to v''.

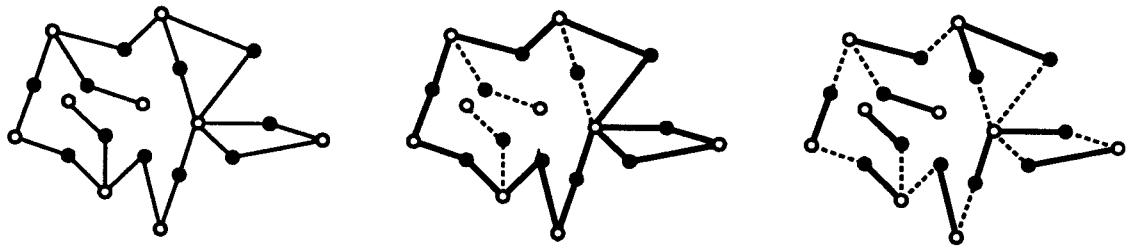
H will be constructed during all steps of the triangulation of the entire graph G. H is planar and bipartite. Call vertices v'' representing the 2-subgraphs V'' *white* and the other vertices of H *black* (see figure 8(a)). We have to find a subset $M \subseteq E_H$ such that every white vertex v' has one incident edge $(v', v) \in M$ and the black vertices have as little as possible incident edges in M. These edges in M correspond with the edges, which will be added to eliminate the multiple edges. To obtain this, we do the following: using a simple modification of Eulers technique to find an Eulerian cycle in a graph, we can extract the elementary cycles C_{elem} from H. An elementary cycle is a cycle that uses each edge at most once. Thus $H - C_{elem}$ consists of paths P with disjoint begin- and endpoints (see figure 8(b)). From C_{elem} and every path P we add alternatingly one edge to M and one not. Recall that H is bipartite, and all white vertices have degree 2. Hence for every vertex in C_{elem} and every internal vertex of a path P, one incident edge e is in M and the other is not. But also for every white vertex, exactly one incident edge is in M, hence satisfying the constraints (see figure 8(c)). For every vertex $v \in H$ with $deg_H(v) \geq 2$, at most $\lceil \frac{deg_H(v)}{2} \rceil$ neighbors are in M.

For each white vertex v'' exactly one incident edge, say (v'', v), is in M. We remove the corresponding multiple edge (w_1, w_2) or (v_{k+1}, v') in G' and we add the edge from a vertex in V'' to v. This can easily be constructed in linear time. This removes all multiple edges and gives a triangulated planar graph G'.

Theorem 4.13 $\Delta(G') \leq \lceil \frac{3}{2}\Delta(G) \rceil + 21$.

Proof: Every vertex v receives at most 2 edges to admit biconnectivity. v receives at most 5 edges when it is new, at most 2 edges when it is left- or rightmost, and at most 5 edges when it is internal. Thus in G v receives at most 14 extra incident edges. In H from every vertex v at most $\lceil \frac{deg'(v)}{2} \rceil$ incident edges are in M, implying $\lceil \frac{deg'(v)}{2} \rceil$ augmenting edges in G to destroy the multiple edges. Since $deg'(v) \leq \Delta(G) + 14$, the theorem follows. □

Theorem 4.14 *G' can be constructed in linear time.*



(a) example of H with white and black vertices.

(b) The elementary cycles and the disjoint paths.

(c) The set M of edges.

Figure 8: The construction of H and M.

Proof: The algorithm BICONNECT can be implemented to run in linear time. Finding the triconnected components can be done in linear time by the well known algorithm of Hopcroft & Tarjan [11], but also by the planar embedding algorithm of Chiba et al. [2]. Both algorithms can easily be modified such that it outputs all 2-subgraphs. From this we can construct the 2-subgraph-tree T in linear time. Computing the canonical 3-ordering for all triconnected components can also be achieved in linear time.

For triangulating a face F we have to unfold the marked edges. Then we have to find the forbidden edges by inspecting the adjacency lists of the vertices of F, except its leftmost and rightmost vertex. If F has n_F vertices, then triangulating F requires $O(n_F)$ time, since $O(n_F)$ edges are added by ZIGZAG. Every vertex is exactly once new and once internal, so we inspect every adjacency list at most twice. The sum of all lengths of all adjacency lists is $O(n)$.

Constructing the graph H can be done in linear time. Finding M in H is equal to finding the elementary cycles and disjoint paths, which can be executed in linear time. Eliminating the multiple edges can be done in linear time by maintaining pointers between v'' in H and V'' in G. □

This completes the proof of theorem 4.1.

This algorithm matches the lower bound by only an additive constant in the biconnected case. If G is already triconnected, then multiple edges cannot occur. In this case by a simple analysis of the algorithm the following theorem can be proved:

Theorem 4.15 *There is a linear algorithm to triangulate a triconnected planar graph G such that for the triangulation G', $\Delta(G') \leq \Delta(G) + 5$ holds.*

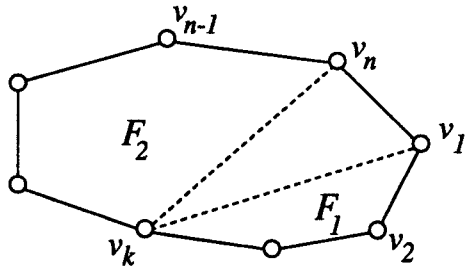


Figure 9: The triangulation of one face while minimizing the maximum degree can recursively be defined.

5 Triangulating one face

In this section we inspect the following problem: given a planar graph G with some embedding, triangulate one face F of G while minimizing the maximum degree. Each vertex v_i of F has degree ≥ 2 and 'forbidden' edges (v_i, v_j), $v_i, v_j \in F$, may occur. We show that by using dynamic programming this problem can be solved exactly in polynomial time.

Let the face F of G on n vertices $v_1, \ldots, v_n$ be given (numbered clockwise around the ring). We notice that in every triangulation the vertices v_1 and v_n have a common neighbor v_k $(2 \leq k \leq n-1)$ inside face F, which splits the face F into two faces F_1 (with vertices $v_1, \ldots, v_k$) and F_2 (with vertices $v_k, \ldots, v_n$). If (v_1, v_k) or (v_k, v_n) is already present outside F, then this triangulation is not possible, since it would imply multiple edges. We recursively triangulate the faces F_1 and F_2. Let F_1' and F_2' denote the triangulated faces, then the highest degree in F_1' and F_2' is important, but moreover, since F_1 and F_2 share v_k, the increase of $deg(v_k)$ in F_1' and F_2' must be added to $deg(v_k)$ in F. When we examine triangulations of a face F_{ij}, formed by vertices $v_i, v_{i+1}, \ldots, v_j$, we inspect the different values of increases of $deg(v_i)$, $deg(v_j)$ and $deg(v_k)$ in F_1 and F_2. See figure 9.

Notice that when $k = 2$ (or $n-1$) then the edge (v_1, v_2) (or (v_{n-1}, v_n)) is already present, hence need not to be added. To deal with this, we delete the edges (v_i, v_{i+1}) $(1 \leq i < n)$ and decrease $deg(v_i)$ by 2. Let $inc(v_i)$ denote the increase of $deg(v_i)$ when triangulating F_1 to F_1' (assuming $v_i \in F_1$). For a triangulation of a face F_{ij} with vertices $v_i, \ldots, v_j$ we have to store the different increases of $deg(v_i)$ and $deg(v_j)$ in a table. Let $D[i, j, i1, j1]$ be the maximum degree of F_{ij} by a triangulation such that $inc(v_i) = i1$ and $inc(v_j) = j1$. If such a triangulation does not exist, $D[i, j, i1, j1] = \infty$. A simple analysis shows the following recursive formulae if $i < j - 1$:

$$D[i,j,i1,j1] := \min \{\max \{ D[i,k,i1-1,k1], D[k,j,k2,j1-1],$$
$$\begin{array}{ll} i<k<j & deg(v_i)+i1+1, \\ k1,k2 & deg(v_j)+j1+1, \\ (i,k) \text{ and } (j,k) & deg(v_k)+k1+k2+2 \\ \text{not forbidden} & \} \end{array}$$
$$\}$$

If $i = j-1$ then for all $i1, j1 \geq 0$: $D[i,j,i1,j1] = \max\{deg(v_i), deg(v_j)\}$. We want to compute $\min_{i1,j1}\{D[1,n,i1,j1], deg(v_1)+i1+1, deg(v_n)+j1+1\}$. We do this by using dynamic programming, based on the above formulaes, and some other ideas which help to decrease the running time of the algorithm.

Theorem 5.1 *There is an exact $O(n^3\Delta(G)\log\Delta(G))$ algorithm to triangulate one face of n vertices of a graph G such that the maximum degree of the triangulation G' is minimized.*

Proof: Let a face F be given. By theorem 4.1, we know that we can triangulate F such that the maximum degree is at most $\lceil \frac{3}{2}\Delta(G) \rceil + 21$. We do not compute all values of $D[i,j,i1,j1]$ as this would be too time-consuming, but instead apply binary search on the maximum degree. So we must test for $O(\log\Delta(G))$ values of K whether a triangulation with maximum degree $\leq K$ exists. Let $D_k[i,j,i1,j1] =$ *true* $\Longleftrightarrow D[i,j,i1,j1] \leq K$. Suppose K is fixed. Note that it is sufficient to know: for all $i1$, $1 \leq i1 \leq K - deg(v_i)$, the smallest value of $j1$ such that $D_k[i,j,i1,j1]$ is *true* and for all $j1$, $1 \leq j1 \leq K - deg(v_j)$, the smallest value of $i1$ such that $D_K[i,j,i1,j1]$ is *true*. Denote these smallest values with $F_K[i,j,i1]$ and $G_K[i,j,j1]$. (If such $j1$ or $i1$ not exist, $F_K[i,j,i1] = \infty$, or $G_K[i,j,j1] = \infty$.)

Now $D_K[i,j,i1,j1] =$ *true*, if and only if there exists a k, $i < k < j$, (v_i, v_k) and (v_j, v_k) not forbidden and $F_K[k,j,K-deg(v_k)-F_K[i,k,i1]-2] \leq j2$. (The increase of the degree of v_k in face F_{kj} may not be larger than $K - deg(v_k) - F_K[i,k,i1] - 2$. $F_K[i,k,i1]$ edges are used for face F_{ik}, also there are edges $(i,k),(k,j)$.) So $F_K[i,j,i1] = \min\{F_K[k,j,K-deg(v_k)-F[i,k,i1]-2] | i < k < j, (i,k),(k,j)$ not forbidden$\}$. The latter formulae allows us to compute all values of $F_K[i,j,i1]$ $(1 \leq i \leq n, 0 \leq i \leq k)$ in $O(n^3K)$ time. When F_K is computed, one easily determines whether a triangulation with maximum degree $\leq K$ exists. Using binary search on K, the total runtime becomes $O(n^3\Delta(G)\log\Delta(G))$. □

This algorithm can be used in combination with the algorithm in section 3 to get an approximation algorithm. We use the canonical 3-ordering as described in section 3 and in each step we add a vertex or face to G_k. Instead of applying the linear approximation algorithm of section 3 we can use this algorithm. Though this seems to give a good approximation, there are inputs (see figure 2(b)), for which this algorithm will imply $\Delta(G') = 2\Delta(G) + O(1)$.

6 Triangulating outerplanar graphs

In this section we inspect the problem of triangulating the internal faces of an outerplanar graph G. An outerplanar graph is a planar graph in which all vertices share one common face, the outerface. When G is biconnected, then the obtained triangulated graph is called a maximal outerplanar graph (or *mop*) and has several properties: a mop has $2n-3$ edges and adding any edge to it destroys the outerplanarity. Every triangulated polygon is a mop [17]. Also recognizing outerplanar graphs is based on recognizing mops [18].

First, we remark that we can treat all 2-edge-connected components separately. The only relevant information for a 2-edge-connected component about the rest of the graph is the number of bridges, adjacent to each vertex in the component, as these count for the degree of the vertex. For each 2-edge-connected component, we make a tree $H = (V_H, E_H)$ with each internal face F_i in this biconnected component represented by a vertex v_{F_i}. We have two types of edges:

- take an edge between two vertices if the corresponding faces share an edge. By outerplanarity, this gives a tree for each set of faces of a biconnected component.
- Make a rooted tree T of the biconnected components of G, such that each child component shares a vertex in G with its father component. Now, for each edge of T, take a face in the father component and a face in the child component that share a vertex, and add an edge in H between the vertices representing these faces.

We now have a tree where each vertex represents a face. Take a node X, representing a face in the biconnected component representing a face in the biconnected component represented by the root of T, and root H at face node X. The *parent-face* of face F_i is the face, represented by the node that is the parent of the node that represents F_i in H. If F_i shares edge (v_a, v_b) (vertex v_a) with its parent-face, then (v_a, v_b) (v_a) is called the *parent-edge* (*parent-vertex*) of F_i. For every edge (v_a, v_b) of face F_i that is a parent-edge of some other face F_j, define $D[v_a, v_b, i1, j1]$ to be the maximum degree if F_j and all their descendants in H are triangulated, such that the degree increase of v_a is at most $i1$, and the degree increase of v_b is most $j1$. For every vertex v_a of face F_i that is parent-vertex of at least one other face F_j, define $D[v_a, a1]$ to be the maximum degree, if these faces F_j and all their descendants in H are triangulated, such that the degree increase of v_a is at most $i1$.

We now use a procedure, similar to the one, described in section 5. Use binary search on the maximum degree K. For a fixed K, define $F_K[v_a, v_b, i1]$ and $G_K[v_a, v_b, j1]$ as in the proof of theorem 5.1. Define $H_K(v_a) = \min\{a1 | D[v_a, a1] \leq K\}$. Compute the tables or values for F_K, G_K and H_K bottom up in the tree H. When we deal with a face, we add $H_K(v_a)$ to the degree of v_a, and let the tables $F_K[v_a, v_b, a1]$ and $G_K[v_a, v_b, b1]$ play the same role as they played in the proof of

theorem 5.1. With minor modifications of this algorithm, one can compute the table $F_K[v'_a, v'_b, a1]$ and $G_K[v'_a, v'_b, b1]$ for the father-edge (v'_a, v'_b) of F_i, or compute the contribution to $H_K[v'_a]$ for the father-vertex v'_a of F_i. We have omitted some easy details here. Applying a similar proof as in theorem 5.1, the following result can be obtained:

Theorem 6.1 *There is an $O(n^3\Delta(G)\log\Delta(G))$ algorithm to triangulate all interior faces of an outerplanar graph while minimizing the maximum degree.*

7 Final Remarks

In this paper we inspected the problem of triangulating a planar graph such that the maximum degree is minimized. It is shown that this problem is NP-complete for biconnected planar graphs. A linear approximation algorithm is presented, working only a constant from the presented lower bounds. This algorithm is heavily based on the canonical 3-ordering for triconnected planar graphs, which is a modification of the canonical ordering of [6]. This ordering can also be determined in linear time and leads to a good approximation algorithm. This comes from the observation that only information of vertices on the outerface is sufficient in every step. This technique also already lead to a linear implementation of the grid drawing algorithm of De Fraysseix et al. [15]. It is interesting to inspect related problems on planar graphs, for which the canonical 3-ordering also can lead to simple linear time approximations, by only maintaining local information in the vertices of the outerface.

This paper also gives more insight in the augmentation problems, which seems to be quite popular nowadays [4, 5, 12, 13, 14, 16, 20]. However, we were not able to prove that the algorithm in this paper works only an additive constant from optimal in the biconnected case. We conjecture that the algorithm in section 4 is only an additive constant worse than optimal, in case the input graph is biconnected. This conjecture is still open and interesting for further study. The NP-completeness result presented in this paper only holds for biconnected planar graphs and is open for triconnected planar graphs. It is interesting to inspect this problem, to come to a more combinatorial insight in planar graphs.

Acknowledgements

The authors wish to thank David Eppstein, for proposing the dynamic programming approach, described in section 5.

References

[1] Booth, K.S., and G.S. Lueker, Testing for the consecutive ones property, interval graphs and graph planarity testing using PQ-tree algorithms, *J. of Computer and System Sciences* 13 (1976), pp. 335–379.

[2] Chiba, N., T. Nishizeki, S. Abe and T. Ozawa, A linear algorithm for embedding planar graphs using PQ-trees, *J. of Computer and System Sciences*, Vol. 30 (1985), pp. 54–76.

[3] Chiba, N., T. Yamanouchi and Nishizeki, Linear algorithms for convex drawings of planar graphs, In: J.A. Bondy and U.S.R. Murty (Eds.), *Progress in Graph Theory*, Academic Press, Toronto, 1984, pp. 153–173.

[4] Eswaran, K.P., and R.E. Tarjan, Augmentation problems, *SIAM J. Comput.* 5 (1976), pp. 653–665.

[5] Frank, A., Augmenting graphs to meet edge-connectivity requirements, *Proc. 31th Annual IEEE Symp. on Found. on Comp. Science*, St. Louis, 1990, pp. 708–718.

[6] Fraysseix, H. de, J. Pach and R. Pollack, How to draw a planar graph on a grid, *Combinatorica* 10 (1990), pp. 41–51.

[7] Fraysseix, H. de, and P. Rosenstiehl, A depth first characterization of planarity, *Annals of Discrete Math.* 13 (1982), pp. 75–80.

[8] Garey, M.R., D.S. Johnson and L. Stockmeyer, Some simplified NP-complete graph problems, *Theoret. Comput. Science* 1 (1976), pp. 237–267.

[9] Haandel, F. van, *Straight Line Embeddings on the Grid*, Dept. of Comp. Science, Report no. INF/SCR-91-19, Utrecht University, 1991.

[10] Hopcroft, J., and R.E. Tarjan, Dividing a graph into triconnected components, *SIAM J. Comput.* 2 (1973), pp. 135–158.

[11] Hopcroft, J., and R.E. Tarjan, Efficient planarity testing, *J. ACM* 21 (1974), pp. 549–568.

[12] Hsu, T., and V. Ramachandran, A linear time algorithm for triconnectivity augmentation, in: *Proc. 32th Annnual IEEE Symp. on Found. on Comp. Science*, Porto Rico, 1991.

[13] Hsu, T., and V. Ramachandran, *On Finding a Smallest Augmentation to Biconnect a Graph*, Computer Science Dept., University of Texas at Austin, Texas, Tech. Rep. TR-91-12, 1991.

[14] Kant, G., *Optimal Linear Planar Augmentation Algorithms for Outerplanar Graphs*, Techn. Rep. RUU-CS-91-47, Dept. of Computer Science, Utrecht University, 1991.

[15] Kant, G., *A Linear Implementation of De Fraysseix' Grid Drawing Algorithm*, Manuscript, Dept. of Comp. Science, Utrecht University, 1988.

[16] Kant, G., and H.L. Bodlaender, Planar graph augmentation problems, Extended Abstract in: F. Dehne, J.-R. Sack and N. Santoro (Eds.), *Proc. 2nd Workshop on Data Structures and Algorithms*, Lecture Notes in Comp. Science 519, Springer-Verlag, Berlin/Heidelberg, 1991, pp. 286–298.

[17] O'Rourke, J., *Art Gallery Theorems and Algorithms*, Oxford Univ. Press, New York, 1987.

[18] Mitchell, S.L., Linear algorithms to recognize outerplanar and maximal outerplanar graphs, *Inform. Process. Lett.* 9 (1979), pp. 229–232.

[19] Read, R.C., A new method for drawing a graph given the cyclic order of the edges at each vertex, *Congr. Numer.* 56 (1987), pp. 31–44.

[20] Rosenthal, A., and A. Goldner, Smallest augmentations to biconnect a graph, *SIAM J. Comput.* 6 (1977), pp. 55–66.

[21] Schnyder, W., Embedding planar graphs on the grid, in: *Proc. 1st Annual ACM-SIAM Symp. on Discr. Alg.*, San Francisco, 1990, pp. 138–147.

[22] Tutte, W.T., Convex representations of graphs, *Proc. London Math. Soc.*, vol. 10 (1960), pp. 304–320.

[23] Woods, D., *Drawing Planar Graphs*, Ph.D. Dissertation, Computer Science Dept., Stanford University, CA, Tech. Rep. STAN-CS-82-943, 1982.