A Note on Computing a Maximal Planar Subgraph using PQ-Trees

Michael Jünger, Sebastian Leipert, Petra Mutzel

Abstract— **The problem of computing a maximal planar subgraph of a non planar graph has been deeply investigated over the last 20 years. Several attempts have been tried to solve the problem with the help of PQ-trees. The latest attempt has been reported by Jayakumar *et al.* [10].**

In this paper we show that the algorithm presented by Jayakumar *et al.* is not correct. We show that it does not necessarily compute a maximal planar subgraph and we note that the same holds for a modified version of the algorithm presented by Kant [12]. Our conclusions most likely suggest not to use PQ-trees at all for this specific problem.

Keywords— **PQ-Trees, Maximal Planar Subgraphs, Planarization.**

I. Introduction

The minimum number of layers needed in the layout of printed circuit boards and integrated chips is equal to the thickness of the interconnection graph [15]. The thickness of a graph G is the minimum number of planar subgraphs whose union is G. In VLSI design the thickness problem is approximated by successively subtracting large planar subgraphs from a given nonplanar graph. Another widely used method in VLSI design and Automatic Graph Drawing is to construct a planar subgraph from a given nonplanar graph by deleting a small number of edges and then to reinsert the removed edges, such that the number of edge crossings is small. However, the problem of finding the minimum number of edges that have to be removed from a given graph in order to obtain a planar subgraph, is known to be an $\mathcal{NP}$-hard problem (see Garey and Johnson [7]).

Therefore, research has focused on computing maximal planar subgraphs. Let $G = (V, E)$ be a simple graph with n vertices and m edges then a planar subgraph G' of G is a *maximal planar* subgraph, if for all edges $e \in G - G'$ the addition of e to G' destroys planarity. Besides a trivial $\mathrm{O}(nm)$ algorithm that can be constructed using any $\mathrm{O}(n)$ planarity test, three different approaches are known for solving this problem.

Chiba, Nishioka and Shirakawa [3] presented an algorithm based on the path addition algorithm that computes a maximal planar subgraph in $\mathrm{O}(nm)$ time. Cai, Han, and Tarjan [2] presented later an $\mathrm{O}(m \log n)$ algorithm that is based on the path addition algorithm as well. Di Battista and Tamassia [4] described an algorithm that checks in $\mathrm{O}(\log n)$ amortized time, whether an edge can be added to G without destroying planarity, obtaining an $\mathrm{O}(m \log n)$ time algorithm as well.

Ozawa and Takahashi [16] have presented an $\mathrm{O}(nm)$ algorithm using the vertex addition algorithm. Jayakumar, Thulasiraman and Swamy [9] showed that in general this algorithm does not determine a maximal planar subgraph. Moreover, the resulting planar subgraph may not even contain all vertices. Jayakumar, Thulasiraman and Swamy [10] presented an algorithm called PLANARIZE that computes a spanning planar subgraph G_p of G in $\mathrm{O}(n^2)$ time. Furthermore, they present an algorithm called MAX-PLANARIZE that augments G_p to a subgraph G' of G by adding additional edges in $O(n^2)$ time. They claim that G' is a maximal planar subgraph of G if G_p (the result of phase 1 of the two phase algorithm) turns out to be biconnected. Kant [12] shows that this algorithm is incorrect, and suggests a modification of the second phase of the algorithm that augments G_p to a maximal planar subgraph of G, even if G_p is not biconnected, maintaining $O(n^2)$ time requirement.

In this article, we will point out a substantial flaw in both the original and the modified two phase algorithm that was not detected previously as well as new mistakes introduced by Kant. In section 2 we give a brief introduction on PQ-trees and the planarity test using this data structure. In section 3 the principle of the planarization algorithm using the PQ-trees is described. In section 4 we show that the algorithm of Jayakumar *et al.* [10] is incorrect giving a detailed description of the major mistake and discuss the attempt of Kant. In the last section we make some concluding remarks.

II. Planarity test using PQ-trees

We assume that the reader is familiar with the basic graph theoretic definitions as mentioned in [8]. A graph is *planar*, if it can be embedded in the plane without any edge crossings. A graph is obviously planar, if and only if its biconnected components are planar. We therefore assume that G is biconnected. The planarity testing algorithm of Lempel, Even and Cederbaum [14] first labels the vertices of G as $1, 2 \ldots, n$ using an *st-numbering* (see [6]). A numbering of the vertices of G by $1, 2 \ldots, n$ is an *st-numbering*, if the vertices "1" and "n" are adjacent and each other vertex j is adjacent to two vertices i and k such that $i < j < k$. The vertex 1 is denoted by s and the vertex n is denoted by t. The st-numbering induces an ori-

Partially supported by DFG-Grant Ju204/7-2, Forschungsschwerpunkt "Effiziente Algorithmen für diskrete Probleme und ihre Anwendungen" and ESPRIT Long Term Research Project Nr. 20244 (ALCOM-IT).

M. Jünger is with the Institut für Informatik, Universität zu Köln, Pohligstr. 1, 50969 Köln, Germany. E-mail: mjuenger@informatik.uni-koeln.de

S. Leipert is with the Institut für Informatik, Universität zu Köln, Pohligstr. 1, 50969 Köln, Germany. E-mail: leipert@informatik.uni-koeln.de

P. Mutzel is with the Max-Planck-Institut für Informatik, Im Stadtwald, 66123 Saarbrücken, Germany. E-mail: mutzel@mpi-sb.mpg.de

entation of the graph, in which every edge is directed from the incident vertex with the higher st-number towards the incident vertex with the lower st-number. From now on we refer to the vertices of G by their st-numbers and call an edge (u, v), with $v < u$, *incoming* edge of v and *outgoing* edge of u.

For $1 \leq k \leq n$, let G_k denote the subgraph of G induced by the vertex set $V_k := \{1, 2, \ldots, k\}$. The graph G'_k arises from G_k as follows: For each edge $e = (u, v)$, where $v \in V_k$ and $u \in V \setminus V_k$, we introduce a virtual vertex u_e with label u and a virtual edge (u_e, v). Let B_k be a planar embedding of G'_k such that all virtual vertices are placed on the outer face. Then, B_k is called a *bush form*. It has been shown by Lempel *et al.* [14] that G is planar, if and only if for every B_k, $k = 1, 2, \ldots, n-1$, there exists a bush form B'_k isomorphic to B_k, such that all virtual vertices in B'_k labeled $k+1$ appear consecutively.

For an efficient computation of B'_k, the PQ-tree technique [1] is applied. The *PQ-tree* T_k corresponding to the bush form B_k is a rooted ordered tree that consists of three types of nodes:
1. Leaves in T_k represent virtual edges in B_k.
2. *P-nodes* in T_k represent cutvertices in B_k.
3. *Q-nodes* represent maximal biconnected components in B_k.

The *frontier* of a PQ-tree is the sequence of all leaves of T_k read from left to right. The frontier of a node X is the sequence of its descendant leaves read from left to right.

Let E_{k+1} denote the set of leaves in T_k that correspond to the virtual vertices labeled $k+1$. A node X is called *full*, if all leaves in its frontier are in E_{k+1}. A node X is *empty*, if its frontier does not contain any leaf of E_{k+1}. Otherwise, X is called *partial*. A node is called *pertinent*, if it is full or partial. The *pertinent subtree* is the smallest connected subtree that contains all leaves of E_{k+1} in its frontier. The root of the pertinent subtree is called *pertinent root*. Two PQ-trees are *equivalent*, if one can be obtained from the other by one or more of the following operations:
1. Permuting the children of a P-node.
2. Reversing the order of the children of a Q-node.

These operations are called *equivalence transformations* and describe *equivalence classes* on the set of all PQ-trees. An equivalence class of PQ-trees corresponds to a class of permutations called the *permissible permutations*.

It has been shown by Booth and Lueker [1] that B'_k exists if and only if T_k can be converted into an equivalent PQ-tree T'_k such that all pertinent leaves appear consecutively in the frontier of T'_k. Booth and Lueker [1] have defined a set of patterns and replacements called *templates* that can be used to reduce the PQ-tree such that the leaves corresponding to edges of the set E_{k+1} appear consecutively in all permissible permutations. To construct T_{k+1} from T_k they first reduce T_k by use of the templates and then replace all leaves corresponding to virtual edges incident to virtual vertices labeled $k+1$ by a P-node, whose children are the leaves corresponding to the incoming edges of the vertex $k+1$ in G.

The planarity testing algorithm now starts with T_1 and constructs a sequence of PQ-trees $T_1, T_2, \ldots$ If the graph is planar, the algorithm terminates after constructing T_{n-1}. Otherwise it terminates after detecting the impossibility of reducing some T_k, $1 \leq k < n$.

III. Principle of an approach for planarization

The basic idea of a planarization algorithm using PQ-trees presented by Jayakumar *et al.* [10] is to construct the sequence of PQ-trees $T_1, T_2, \ldots, T_{n-1}$ by deleting an appropriate number of pertinent leaves every time the reduction fails such that the resulting PQ-tree becomes reducible. In every step of the algorithm PLANARIZE, a maximal consecutive sequence of pertinent leaves is computed by using a *$[w, h, a]$-numbering* (see [10]). All pertinent leaves that are not adjacent to the maximal pertinent sequence are removed from the PQ-tree in order to make it reducible. Hence the edges corresponding to the leaves are removed from G and the resulting graph G_p is planar.

It has been shown by Jayakumar *et al.* [10] that the graph G_p computed by PLANARIZE is not necessarily maximal planar. The authors therefore suggest to apply a second phase called MAX-PLANARIZE, also based on PQ-trees. Knowing which edges have been removed from G to construct G_p, edges from $G - G_p$ are added back to G_p in the second phase without destroying planarity.

During the reduction of a vertex v, there may exist nonpertinent leaves that are between a pertinent leaf l_v and its maximal pertinent sequence in all permissible permutations of the PQ-tree T_{v-1}. This maximal pertinent sequence has been determined with the help of the $[w, h, a]$-numbering. In order to make the tree T_{v-1} reducible, the leaf l_v is removed from the tree and the corresponding edge is removed from the graph G, guaranteeing that the subgraph G_p will be planar. However, it may occur that the nonpertinent leaves that are positioned between l_v and its maximal pertinent sequence in T_{v-1}, are removed as well from a tree T_k, $v \leq k < n$, in order to obtain reducibility. Therefore, there is no need to remove the edge corresponding to l_v from the graph G.

In order to find leaves such as l_v, Jayakumar *et al.* [10] use the algorithm MAX-PLANARIZE. In step i, both PLANARIZE as well as MAX-PLANARIZE reduce the same vertex i. The difference between the PQ-trees in the two algorithms is, according to the authors, that all leaves that have been deleted in PLANARIZE are ignored in MAX-PLANARIZE from the moment they are introduced into the tree until they get pertinent. This causes the nonpertinent leaves between the pertinent leaf l_v and its maximal pertinent sequence to be ignored. Hence l_v is adjacent to its maximal pertinent sequence and the corresponding edge can be added back to G_p, while the leaves between l_v and the maximal pertinent sequence are removed from the PQ-tree.

IV. On the incorrectness of the algorithm

While some incorrect facts of the approach of Jayakumar *et al.* have been described in a technical report by Kant [12], who attempted to correct the algorithm, a major problem has not been detected.

Jayakumar *et al.* assume that the maximal planar subgraph G_p is biconnected for the correct application of the Lempel-Even-Cederbaum algorithm. Furthermore, as they have stated correctly, this is necessary in order to have an st-numbering. Nevertheless, the PQ-trees in MAX-PLANARIZE are constructed according to the st-numbering that was computed for the graph G.

As a matter of fact, the st-numbering of G does not imply an st-numbering of any subgraph G_p even if the subgraph G_p is biconnected. This results in two problems, of which one is crucial and cannot be dealt with even by the ideas described by Kant [12].

Both problems are based on the fact that during the application of PLANARIZE for some vertices of V all incoming edges may be deleted from the graph while the resulting graph G_p stays biconnected.

Let $v \in V$ be such a node with no incoming edges in G_p. Since G_p is biconnected, v must have at least two outgoing edges (v, u_1) and (v, u_2). Let $w \in V$ be a vertex in G such that $u_1, u_2 < w < v$. Thus the leaves corresponding to the outgoing edges of w are reduced before the leaves of v. Let T_{w-1} be the PQ-tree during the application of MAX-PLANARIZE, in which the relevant leaves corresponding to the outgoing edges of w have to be reduced. Assume that the leaves of both nodes w and v are on the outer face of the same biconnected component of the bush form that corresponds to the PQ-tree T_{w-1}. Assume further that one designated leaf w_{k+1} of the vertex w is separated by the leaves v_1 and v_2 corresponding to (v, u_1) and (v, u_2) from the leaves $w_1, w_2, \ldots, w_k$, where the latter form the maximal pertinent sequence (see Figure 1 for an illustration).

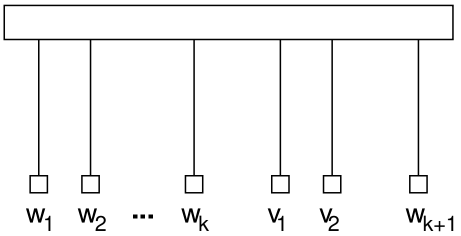

Fig. 1. Leaf w_{k+1} is separated by v_1 and v_2 from its maximal pertinent sequence $w_1, w_2, \ldots, w_k$.

If (v, u_1) and (v, u_2) are the only outgoing edges of v in G_p, then the leaves v_1 and v_2 will be changed during the reduction of the PQ-tree T_{v-1} into a P-node with leaves corresponding to edges in $E \setminus E_p$. If none of the incoming edges of v is added to G_p in a PQ-tree T_i, $v < i < n$, the edge corresponding to the leaf w_{k+1} can be added to the graph G_p without destroying planarity. Hence, the resulting graph G_p is not a maximal planar subgraph.

We now consider the second problem. The planarization algorithm of Jayakumar *et al.* [10] does not obey an important invariant implied by the following lemma, shown by Even [5].

Lemma IV.1: Let $G = (V, E)$ be a planar graph with an st-numbering and let $1 \leq k \leq n$. If the edge (t, s) is drawn on the boundary of the outer face in an embedding of G, then all vertices and edges of $G - G_k$ are drawn in the outer face of the plane subgraph G_k of G.

This result allowed Lempel *et al.* to transform the problem of planarity testing to the construction of a sequence of bush forms B_k, $1 \leq k \leq n$. For a planar graph G, edges and vertices that have not been introduced into the current subgraph G_k are always embedded into the outer face of G_k.

The approach of Jayakumar *et al.* [10] does not obey this invariant in the second phase. There exist edges that have to be embedded into an inner face of some G_k, even if (t, s) is drawn on the outer face. Due to the above lemma, the correction step MAX-PLANARIZE only considers edges for reintroduction into the planar subgraph G_p that are on the outer face of the current graph G_k. Since the numbering that is used to determine the order in which the vertices are reduced does not correspond to an st-numbering of G_p in general, the algorithm of Jayakumar *et al.* [10] ignores edges that can be added into an inner face of the embedding of a current graph G_k without destroying planarity. This fact is fatal, as we are about to show now.

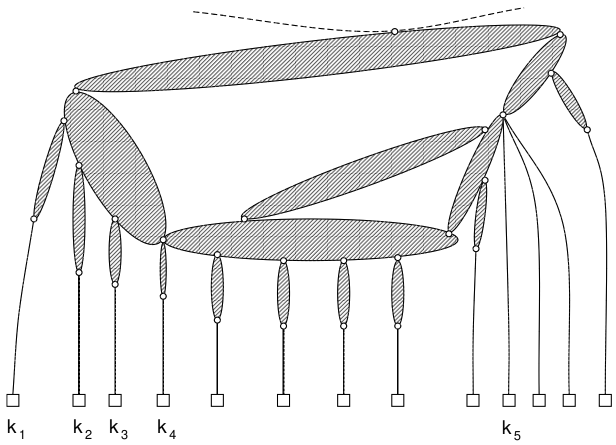

Fig. 2. Part of a bush form B_{k-1}

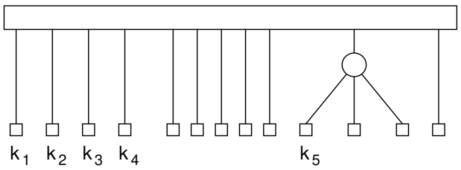

Fig. 3. Part of a PQ-tree corresponding to bush form B_{k-1}

In Figure 2, a part of a bush form B_{k-1}, $1 < k \leq n$, of a graph G is shown. The virtual vertices corresponding to the vertex k are labeled $k_1, k_2, \ldots, k_5$ and all other virtual vertices are left unlabeled. The corresponding part of the PQ-tree is shown in Figure 3. Obviously, there do not exist any reversions or permutations such that the virtual vertices of k occupy consecutive positions. Hence, the graph G is not planar. Applying the $[w, h, a]$-numbering of Jayakumar *et al.* [10] allows us to delete the virtual vertex

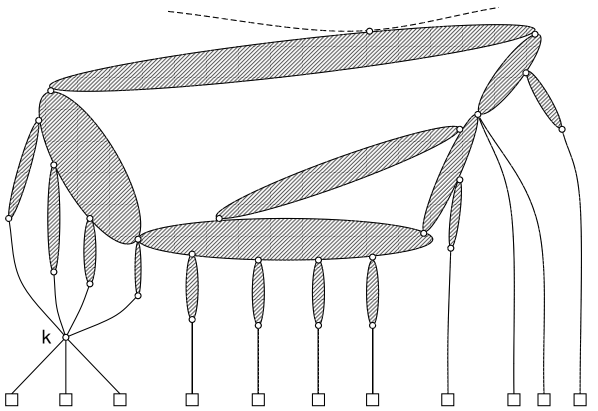

Fig. 4. Part of a bush form B_k

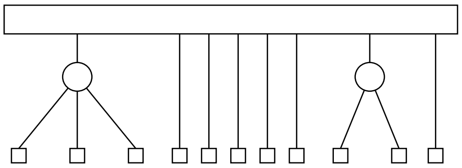

Fig. 5. Part of a PQ-tree corresponding to bush form B_k

k_5 and to reduce the other four vertices k_1, k_2, k_3, k_4. The resulting bush form B_k is planar and the relevant part is shown in Figure 4. Figure 5 shows the corresponding part of the PQ-tree. Assume now that all descendants of k have to be removed from the PQ-tree in a later step. Hence all incoming edges incident on k are removed from the tree. Now assume further that there exists a path $v_1, v_2, \ldots, v_l$ in G_p such that

- for all i, j, $1 \leq i < j \leq l$ the inequality $v_i < v_j$ holds,
- the edge (v_2, v_1) corresponds to one of the leaves that are between the leaf k_5 and the maximal pertinent sequence k_1, k_2, k_3, k_4 in all PQ-trees equivalent to T_{k-1},
- $v_l = t$.

This path guarantees that all outgoing edges of the vertex k cannot be embedded into the outer face of the embedding of B_{k-1} without crossing an edge on this path. Hence the edge e_{k_5} corresponding to the leaf k_5 is not considered by the algorithm MAX-PLANARIZE as being an edge that does not destroy planarity. Therefore, e_{k_5} is not added back to the planar subgraph G_p.

Nevertheless adding the edge e_{k_5} to G_p may not destroy planarity of G_p as is shown in our example in Figure 6. Since all incoming edges of the vertex k have been deleted by PLANARIZE and are not added back by MAX-PLANARIZE, it may be possible to swap the vertex k into an inner face of the embedding of B_k such that the virtual vertex k_5 can be identified with k and the edge e_{k_5} is embedded into the bush form B_k without destroying planarity.

Therefore, the strategy of using PQ-trees presented by Jayakumar *et al.* [10] does not compute a maximal planar subgraph in general. Furthermore, we point out that the same problem holds for the modified version of this algorithm, presented by Kant [12]. This version follows a

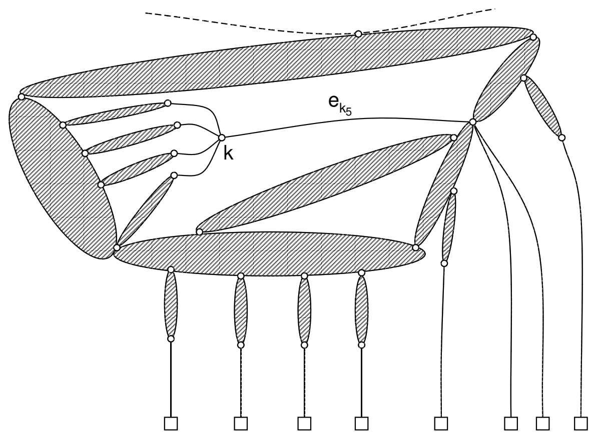

Fig. 6. Part of a bush form B_k with e_{k_5} embedded

similar strategy of computing a spanning planar subgraph G_p using PLANARIZE and then adding edges that do not destroy planarity in a second phase. The order of reductions that is used to insert vertices into existing bush forms is the same as the one implied by the st-numbering on G. Hence this approach is not able to compute a maximal planar subgraph for the same reason.

Summarizing, we state the following lemma.

Lemma IV.2: Let $G = (V, E)$ be a nonplanar graph. Let $G_p = (V, E_p)$, $E_p \subseteq E$, be a planar subgraph of G, such that G_p was obtained from G by
1. computing an st-numbering for all vertices and
2. applying the algorithm of Lempel *et al.* [14] constructing a sequence of bush forms B_k, $1 \leq k \leq n$, by embedding a maximal number of outgoing edges of a vertex k, $1 < k \leq n$, in the outer face of B_{k-1} without crossings, deleting all other outgoing edges of k.

Let $G'_p = (V, E'_p)$, be a planar subgraph of G such that
1. $E_p \subseteq E'_p \subseteq E$,
2. the graph G'_p is computed by constructing a sequence of bush forms B'_k, $1 \leq k \leq n$, based on the st-numbering used for determining G_p, and possibly embedding outgoing edges $e \in E \setminus E_p$ of every vertex k, $1 < k \leq n$, without crossings in the outer face of B_{k-1}.

Then the subgraph G'_p is not necessarily maximal planar.

Proof: Clear from the discussion above. ■

Considering a computation of an st-numbering for the planar subgraph G_p in order to augment G_p to a maximal planar subgraph of G and then constructing a sequence of bush forms B'_k, $1 \leq k \leq n$, is aggravated by the fact that the graph G_p is not biconnected in general. Furthermore, the difference between the bush forms of the first phase and the second phase may result in the deletion of the edges of G_p as soon as edges of $E \setminus E_p$ are added to G_p. Adding an edge $e \in E \setminus E_p$ to G_p is able to change the corresponding bush form in such a way, that the pertinent leaves corresponding to the outgoing edges of some node v in E_p cannot form a consecutive sequence in any permissible permutations.

If the st-numbering of G is as well an st-numbering of G_p, the counterexamples given by Kant [12] show that MAX-

PLANARIZE is not even correct for this special case. However, as has been shown in [13], [11], the modifications suggested by Kant [12] do not correct the algorithm presented by Jayakumar *et al.* [10].

Kant [12] suggested a correction of the second phase by introducing sequence indicators and by delaying the decision, whether a deleted leaf can be added back to G_p, until enough information is available. As has been laid out in [13], [11], this approach does not succeed for three main reasons.

1. If there are several deleted leaves corresponding to the outgoing edges of some vertex $v \in V$, the approach considers more edges for reintroduction than can actually be added without destroying planarity.
2. If edges are added back into the graph G_p, the set of permissible permutations of the corresponding PQ-tree is not restricted in a proper way.
3. If there are several deleted leaves corresponding to the outgoing edges of different vertices that can be added back to G_p without destroying planarity, the approach does not consider all edges for reintroduction. The algorithm therefore does not necessarily compute a maximal planar subgraph.

The first two problems have been shown to be solvable by Leipert [13], but the last still remains unsolved.

V. Concluding Remarks

In this paper we showed that the attempt of Jayakumar *et al.* [10] to solve the maximal planar subgraph problem with PQ-trees is not correct. The problem is due to the fact that an important invariant for planarity testing is ignored. We have further noted that even a corrected version of the algorithm applied in the best possible case, where the st-numbering of a graph G is as well an st-numbering of the planar subgraph G_p, is not correct.

Since this best case is a very rare case and since the modifications for the solved problems (see [13]) are far beyond any reasonable implementation, we doubt that a useful algorithm based on the strategy presented by Jayakumar *et al.* [10] can be found.

References

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

[2] J. Cai, X. Han, and R. E. Tarjan. An $O(m \log n)$-time algorithm for the maximal planar subgraph problem. *SIAM Journal of Comput.*, 22:1142–1162, 1993.

[3] T. Chiba, I. Nishioka, and I. Shirakawa. An algorithm for maximal planarization of graphs. In *Proceedings on the 1979 IEEE International Symposium on Circuits and Systems*, pages 336–441, 1979.

[4] G. Di Battista and R. Tamassia. Incremental planarity testing. In *Proceedings on the 30th Annual IEEE Symposium on Foundations of Computer Science, North Carolina*, pages 436–441, 1989.

[5] S. Even. *Graph Algorithms.* Computer Science Press, Potomac, Maryland, 1979.

[6] S. Even and R. E. Tarjan. Computing an st-numbering. *Theoretical Computer Science*, 2:339–344, 1976.

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* Freeman & Co., San Francisco, 1979.

[8] F. Harary. *Graph Theory.* Addison Wesley, 1969.

[9] R. Jayakumar, K. Thulasiraman, and M.N.S. Swamy. On maximal planarization of non-planar graphs. *IEEE Transactions on Circuits Systems*, 33(8):843–844, 1986.

[10] R. Jayakumar, K. Thulasiraman, and M.N.S. Swamy. On $O(n^2)$ algorithms for graph planarization. *IEEE Transactions on Computer-Aided Design*, 8(3):257–267, March 1989.

[11] M. Jünger, S. Leipert, and P. Mutzel. On computing a maximal planar subgraph using PQ–trees. Technical Report 96.227, Institut für Informatik, Universität zu Köln, 1996.

[12] G. Kant. An $O(n^2)$ maximal planarization algorithm based on PQ-trees. Technical Report RUU-CS-92-03, Department of Computer Science, Utrecht University, 1992.

[13] S. Leipert. Berechnung maximal planarer Untergraphen mit Hilfe von PQ-Bäumen. Master's thesis, Institut für Informatik der Universität zu Köln, 1995.

[14] A. Lempel, S. Even, and I. Cederbaum. An algorithm for planarity testing of graphs. In *Theory of Graphs: International Symposium: Rome, July 1966*, pages 215–232. Gordon and Breach, New York, 1967.

[15] T. Lengauer. *Combinatorial Algorithms for Integrated Circuit Layout.* Wiley, 1990.

[16] T. Ozawa and H. Takahashi. A graph-planarization algorithm and its application to random graphs. In *Graph Theory and Algorithms*, volume 108 of *Lecture Notes in Computer Science*, pages 95–107, 1981.