I N F O R M A T I K

A linear time heuristic for the branch-decomposition of planar graphs

Hisao Tamaki

MPI–I–2003–1-010 March 2003

FORSCHUNGSBERICHT RESEARCH REPORT

M A X - P L A N C K - I N S T I T U T
F Ü R
I N F O R M A T I K

Stuhlsatzenhausweg 85 66123 Saarbrücken Germany

Authors' Addresses

Hisao Tamaki
Meiji University
Department of Computer Science
Kawasaki 214-8517, Japan
tamaki@cs.meiji.ac.jp

Abstract

Let G be a biconnected planar graph given together with its planar drawing. A *face-vertex walk* in G of length k is an alternating sequence $x_0, \ldots x_k$ of vertices and faces (i.e., if x_{i-1} is a face then x_i is a vertex and vice versa) such that x_{i-1} and x_i are incident with each other for $1 \leq i \leq k$. For each vertex or face x of G, let α_x denote the length of the shortest face-vertex walk from the outer face of G to x. Let α_G denote the maximum of α_x over all vertices/faces x. We show that there always exits a branch-decomposition of G with width α_G and that such a decomposition can be constructed in linear time. We also give experimental results, in which we compare the width of our decomposition with the optimal width and with the width obtained by some heuristics for general graphs proposed by previous researchers, on test instances used by those researchers. On 56 out of the total 59 test instances, our method gives a decomposition within additive 2 of the optimum width and on 33 instances it achieves the optimum width.

Keywords

Branch-decomposition, carving-decomposition, planar graph, heuristic, linear time

1 Introduction

Let G be a graph with vertex set $V(G)$ and edge set $E(V)$. A *branch-decomposition* [6] of G is a ternary tree (i.e., a tree in which each non-leaf node has degree exactly 3), in which each edge of G appears as a leaf exactly once. To avoid confusion, we will speak of nodes of branch-decompositions, while we speak of vertices of graph G.

Let B be a branch-decomposition of G. For each ordered pair (p, q) of adjacent nodes in B, we denote by $B_{p,q}$ the subtree of B induced by the nodes reachable from p, including p, without using the edge between p and q. For each subtree B' of B, let $L(B')$ denote the set of edges of G contained, as leaves, in B'. Then, $\{L(B_{p,q}), L(B_{q,p})\}$ is a *separation* of $E(G)$, i.e., $L(B_{p,q}) \cap L(B_{q,p}) = \emptyset$ and $L(B_{p,q}) \cup L(B_{q,p}) = E(G)$. We say that the edge between p and q *induces* this separation. The *width* of a separation $\{S, E(G) \setminus S\}$ of $E(G)$ is the number of vertices of G that are incident to some edge of S and at the same time incident to some edge of $E(G) \setminus S$. The width of a branch-decomposition is the maximum width of the separation induced by an edge of the decomposition. Finally, the branch-width of G is the minimum width over all branch-decompositions of G.

Given a branch-decomposition of G, dynamic programming algorithms for numerous problems can be designed that run in time linear in the size of the graph and exponential in the width of the decomposition. These problems include, to name a few, coloring, vertex cover, Steiner tree and the traveling salesman problem. Indeed, the branch-width of a graph G is linearly related to the tree-width [5] of G and therefore all the problems solvable in linear time on graphs with bounded tree-width (see [1, 2] for example) can be solvable in linear time on bounded branch-width graphs. Therefore, it is of extreme interest to recognize graphs of small branch-width and to find a branch-decomposition of small width for a given graph. Though the problem of deciding the width of a given general graph is NP-complete, it can be solved in $O(n^2)$ time for planar graphs by the algorithm of Seymour and Thomas [7]. Their algorithm can also be used to find a branch-decomposition with the optimal width, but the running time for this task is $O(n^4)$. Therefore, for large graphs, we still need to resort to heuristics even if the graph is planar.

Two effective heuristics for the branch-decomposition of general graphs are known. One is called the eigenvector method due to Cook and Seymour [3] and the other the diameter method due to Hicks [4]. Both methods proceed recursively, building a binary tree of decomposition (which can in the end be viewed as a ternary tree disregarding the root). At each step of recursion, we have a set of edges E and try to decompose this set in two sets E_1 and E_2 so that the widths of the separation $\{E_1, E(G) \setminus E_1\}$ and of $\{E_2, E(G) \setminus E_2\}$ are both small. We also wish to avoid a decomposition that would result in a large width in the further recursive steps. The two methods differ in the manner they try to achieve this second goal. As their names suggest, one uses the eigenvectors of the adjacency matrix, and the other the diameter, of the local graph induced by E to guide the separation step. Though these heuristics have been successful to some extent, the running time observed by experiments has a rather high complexity with respect to the input size and no theoretical upper bound is known.

In this paper, we introduce a heuristic specialized to planar instances. This heuristic is completely different from the above general heuristics and crucially depends on the planarity of the instance. The resulting algorithm is extremely simple and the running time is linear in the size of the input.

Formally, the following theorem captures our heuristic. Let G be a planar graph drawn in the plane. A *face-vertex walk* of G is a sequence $x_0, x_1, \ldots, x_k$ of faces and vertices of G such that, for $1 \leq i \leq k$,

1. if x_{i-1} is a face then x_i is a vertex and vice versa, and
2. x_{i-1} and x_i are incident with each other.

We say that face-vertex walk $x_0, x_1, \ldots, x_k$ is of length k and is from x_0 to x_k. The *face-vertex height* of a vertex or face x of G is the length of the shortest face-vertex walk from the outer face of G to x. Let α_G denote the maximum face-vertex height of a vertex or face in G.

Theorem 1.1 *Given a biconnected planar graph G with a planar drawing, a branch-decomposition of G with width at most α_G can be constructed in linear time.*

The width of the decomposition we obtain from this theorem clearly depends on the choice of the outer face. For some geometric instances, it appears that the outer face in the natural drawing is usually the best choice for minimizing α_G. In general, we may need to look for a good choice of the outer face at the cost of increased running time.

To measure the effectiveness of our heuristic, we perform experiments on planar graph instances taken from [4]. These are delaunay triangulations of points sets which are in turn taken from TSPLIB, the benchmark library for the traveling salesman problem [10]. Experimental data of the two methods for general graphs mentioned above on these instances are available in [4], with which we compare our results. The quality of our decompositions is similar to that produced by the diameter method and on the whole better than the eigenvector method. On the other hand, our method has a great advantage in terms of the running time. It is typically three orders of magnitude faster than the other methods and for some instances as much as 10^6 times faster (though this must be discounted somewhat by the difference in the CPU speed).

We also compare the width of our decompositions with the optimal width, which was computed by Hicks [4] using his variant of the exact algorithm in [7]. On 56 out of the total 59 test instances, our decomposition is within additive 2 of the optimum width and on 33 instances it achieves the optimum width.

These experimental results suggest that our heuristic exploits planarity extremely well in finding good branch-decompositions.

The rest of this paper is organized as follows. In the next section, we describe the notion of carving-decompositions of a graph, which is closely related to that of branch-decompositions, and state a theorem on carving-decompositions which implies our main theorem. In Section 3, we introduce a certain type of spanning trees of a planar graph, which we call medial-axis trees, and study its properties. In Section 4, we prove the theorem on carving-decompositions using a construction based on a medial-axis tree. Section 5 contains some experimental results. We mention some applications and future research directions in the conclusion section.

2 Carving- and branch-decompositions

In this section, we describe the notion of carving-decompositions and state their relation to branch-decompositions. Our theorem on branch-decompositions stated in the introduction is a consequence of another theorem on carving-decompositions stated in this section.

Let G be a graph. A *carving-decomposition* [6] of G is defined similarly to a branch-decomposition, but differs in that it is a decomposition of $V(G)$ rather than of $E(G)$. More formally, a *carving-decomposition* of G is a ternary tree in which each vertex of G appears as a leaf exactly once. Let C be a carving-decomposition of G. For each ordered pair (p, q) of adjacent nodes in B, we denote by $C_{p,q}$ the subtree of C induced by the nodes reachable from p, including p, without using the edge between p and q. For each subtree C' of C, let $L(C')$

denote the set of vertices of G contained, as leaves, in C'. We say that the edge between p and q induces the separation $\{L(C_{p,q}), L(C_{q,p})\}$ of $V(G)$. We also say that this separation is *in* C. The *cut* associated with a separation $\{S, V(G) \setminus S\}$ of $V(G)$ is the set of edges each having one end in S and the other end in $V(G) \setminus S$. The *width* of a separation of $V(G)$ is the cardinality of its associated cut. The width of a carving-decomposition is the maximum width over all the separations in the decomposition. The *carving-width* of G is the minimum width over all the possible carving-decompositions of G.

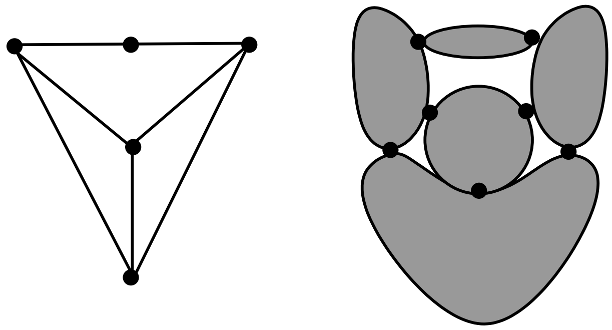
In the rest of this paper, we are only concerned with planar graphs. When we speak of a planar graph G, we assume that a planar drawing of G is given together. When we consider the branch-decomposition of G, we will assume that G is biconnected, since otherwise the branch-decomposition of G can be obtained by combining the branch-decompositions of the biconnected components of G. We will moreover assume that G does not contain parallel edges, as they do not affect the branch-width. However, we will need to consider graphs with parallel edges when we consider carving-decompositions, especially in the following context of relating the branch- and carving-decompositions. We will always assume that our graphs do not contain self-loops.

Let G be a planar graph. The *medial graph* [7] of G, denoted by $M(G)$ is constructed as follows. $M(G)$ has a distinct vertex x_e for each edge e of G. It has an edge between x_e and $x_{e'}$ if and only if e and e' share an endvertex v and they are consecutive in the counterclockwise order around v. When e and e' are the only edges incident with v, then there are two parallel edges between x_e and $x_{e'}$. It is not difficult to see that $M(G)$ is planar: draw each x_e at the bisection point of the drawing of e and for each vertex v of G, with edges $e_1, \dots, e_k$ incident with it in counterclockwise order, draw a closed curve around v going through $x_{e_1}, \dots, x_{e_k}$ in this order. See Figure 2 for an example. Note that there are two types of faces of $M(G)$: one type corresponds to a vertex of G and the other corresponds to a face of G. In Figure 2, the faces corresponding to vertices of G are shaded. A face of one type is incident (through an edge) only with faces of the other type, i.e., the dual of $M(G)$ is bipartite. Note, however, that this observation is valid only under our convention that G itself does not contain parallel edges, even though $M(G)$ may.

Seymour and Thomas [7] show that the carving-width of $M(G)$ is exactly twice the branch-width of G. We only need the following lemma to this theorem.

Lemma 2.1 (Seymour and Thomas [7]) *Let C be a carving-decomposition of $M(G)$ with width w. Then G has a branch-decomposition of G with width*

Figure 1: Planar graph G and its medial graph



at most $w/2$.

Proof: By the one-to-one correspondence between $E(G)$ and $V(M(G))$, a branch-decomposition B of G is naturally induced by the carving-decomposition C of $M(G)$. Let $\{X, \bar{X}\}$ be an arbitrary separation in C and let $\{E, \bar{E}\}$ be the corresponding separation in B, i.e., $X = \{x_e \mid e \in E\}$. Let U be the set of vertices of G each incident to both E and $\bar{E}$. Then, for each $v \in U$, C must contain at least two edges from the cycle bounding the face of $M(G)$ corresponding to v. Therefore, the width of $\{X, \bar{X}\}$ is at least twice the width of $\{E, \bar{E}\}$. It follows that the width of C is at least twice the width of B. $\square$

Our main theorem is obtained by applying the following theorem on carving-decomposition to the medial graph.

Let G be a planar graph. We denote by G^* the planar dual of G. To each vertex v, edge e and face f of G correspond a face v^*, an edge e^* and a vertex f^* of G^*, called duals of v, e and f, respectively. For each vertex or face x of G and each edge e of G, x^* is incident with e^* in G^* if and only if x is incident with e in G.

For each face f of G, the *height* of f, denoted by $h(f)$, is defined to be the length of the shortest path in G^* from f^*_{out} to f^*, where f_{out} is the outer face of G and the length of a path is the number of edges in the path.

Let v be a vertex of G. For two distinct faces f, g incident to v, let $\pi^*_v(f, g)$ denote the dual path from f^* to g^* counterclockwise around the dual face v^*. Define $\beta_v(f, g)$ by $\beta_v(f, g) = |\pi^*_v(f, g)| + h(f) + h(g)$, where $|\cdot|$ denotes the length of the path. Then define β_v for each $v \in V(G)$ by:

$$\beta_v = \min_{f \neq g}\{\max\{\beta_v(f, g), \beta_v(g, f)\}\},$$

where the minimum is taken over all distinct pairs f, g of faces incident with v. Finally, we define β_G to be the maximum of β_v over all $v \in V(G)$. We also denote by δ_G the maximum vertex degree of G.

Theorem 2.2 *Given a planar graph G, a carving-decomposition of G with width at most* $\max\{\beta_G, \delta_G\}$ *can be constructed in linear time.*

We need the following observation in order to obtain Theorem 1.1 from this theorem.

Proposition 2.3 *Let G be a biconnected planar graph and let $M(G)$ be the medial graph of G. Then $\beta_{M(G)} \leq 2\alpha_G$.*

Proof: Let v be an arbitrary vertex of $M(G)$. Note that every vertex of $M(G)$ has degree 4. Let f_1, f_2, f_3, f_4 be the faces incident with v in counterclockwise order and, without loss of generality, assume that $h(f_1)$ is the largest of $h(f_i)$, $1 \leq i \leq 4$. Since the dual of $M(G)$ is bipartite, two adjacent faces cannot have the same height and hence we have $h(f_2) = h(f_4) = h(f_1) - 1$. Therefore, we have $\beta_v(f_2, f_4) = \beta_v(f_4, f_2) = 2(h(f_1) - 1) + 2 = 2h(f_1)$ and hence $\beta_v \leq 2h(f_1)$. But $h(f_1)$ equals the face-vertex height of the vertex or face of G corresponding to f_1. It follows that we have $\beta_v \leq 2\alpha_G$ for every vertex v of $M(G)$. □

Since any carving-decomposition of $M(G)$ with width $2w$ straightforwardly translates, in linear time, into a branch decomposition of G with width w through Lemma 2.1, Theorem 1.1 is an immediate consequence of Theorem 2.2 and Proposition 2.3.

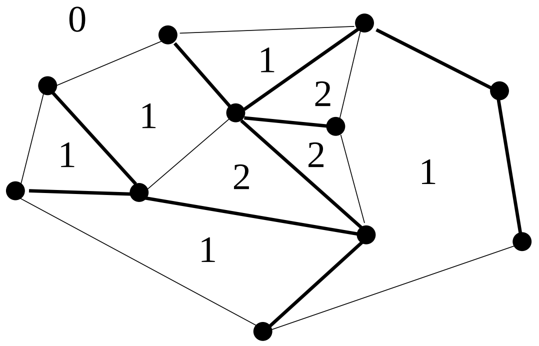
We prove Theorem 2.2 in the following two sections.

3 Medial-axis tree

Let G be a planar graph. Throughout this and the next section, we assume that G is 2-edge-connected. Note that this is true for medial graphs. In general, this assumption is not essentially restrictive either, since the problem of finding a good carving decomposition of general planar graph G trivially reduces to that of finding good carving decompositions of the 2-edge-connected components of G.

Our carving-decomposition of G is based on a certain spanning tree of G which we call a *medial-axis tree* of G. Let f_{out} denote the outer face of G, and let

Figure 2: A medial-axis tree of a planar graph: bold lines indicate tree edges and the number in each face indicates its height



T^* be a breadth-first tree in G^*, the dual of G, rooted at f^*_{out}. That is, each path in T^* from the root to some vertex f^* is the shortest path from f^*_{out} to f^* in G^*.

Let T be the spanning tree of G dual to T^*, i.e., T is a spanning subgraph of G such that an edge e of G is in T if and only if e^* is not in T^*. Since the connectedness of T is equivalent to the acyclicity of T^* and vice versa, T is indeed a spanning tree of G. We call T a medial-axis tree of G. This name is in analogy of the geometric notion: the *medial-axis* of a bounded region in the plane is the locus of points p such that there are at least two points on the boundary of the region that minimize the distance to p over all the points on the boundary. The property of each edge e of T that the two faces incident with e are almost equidistant in the dual graph to the outer face (the distances differing by at most 1) supports this analogy, though somewhat in a loose manner. Given a drawing of G, it is clear that a medial-axis tree of G can be constructed in linear time.

Fix a medial-axis tree T of G and let v be a vertex of G. Let $e_1, \ldots, e_k$ be the edges incident to v listed in the counterclockwise order. Between edges e_i and e_{i+1} (the latter should be interpreted as e_1 when $i = k$), v may have more than one incident face but the face with the minimum height is unique, since the dual of these faces belong to a path of T^* and there is a unique vertex of this dual path closest to the root. We denote this face of minium height between e_i and e_{i+1} by f_i and call $f_1, \ldots, f_k$ the *delimiting faces* of v.

For each distinct pair f, g of delimiting faces of v, let $E_v(f, g)$ denote the set of edges of T incident to v that appear counterclockwise after f and before g around v. In the notation of the previous paragraph, $E_v(f_i, f_j) = \{e_h \mid i < h \leq j\}$ if $i < j$.

For each vertex v of G and a distinct pair f, g of its delimiting faces, let

$T_v(f,g)$ denote the subtree of T consisting of v and all vertices reachable from v through a path of T containing some edge of $E_v(f,g)$. We call such a subtree a *contiguous subtree* of T at v specified by f and g.

We are interested in analyzing the separations of the form $\{V(T'), V(G) \setminus V(T')\}$, where T' is a contiguous subtree of T, since they appear in the carving-decomposition we are going to construct.

Let f and g be distinct delimiting faces of v. Let $\lambda_v^*(f,g)$ denote the path in T^* that goes from f^* up to the lowest common ancestor of f^* and g^* and then down to g^*. Draw a closed curve γ through v that stays within the faces of G corresponding to the vertices of $\lambda_v^*(f,g)$ and crosses, exactly once, each edge of G corresponding to an edge of $\lambda_v^*(f,g)$. Observe that this curve separates the subtree $T_v(f,g)$ from the other parts of T. More precisely, any edge of G between a vertex of $T_v(f,g)$ and a vertex not in $T_v(f,g)$ must either be incident with v or cross γ. Edges in the first category correspond to the dual edges in $\pi_v^*(g,f)$ in the notation of Section 2. Edges in the second category correspond to the edges in $\lambda_v^*(f,g)$. Since the length of $\lambda_v^*(f,g)$ is at most $h(f)+h(g)$, we have the following lemma.

Lemma 3.1 *The width of separation $\{V(T_v(f,g)), V(G) \setminus V(T_v(f,g))\}$ in G is at most $\beta_v(g,f)$.*

4 Constructing the carving-decomposition

We are now ready to describe the construction of our carving-decomposition that satisfies the claim of Theorem 2.2.

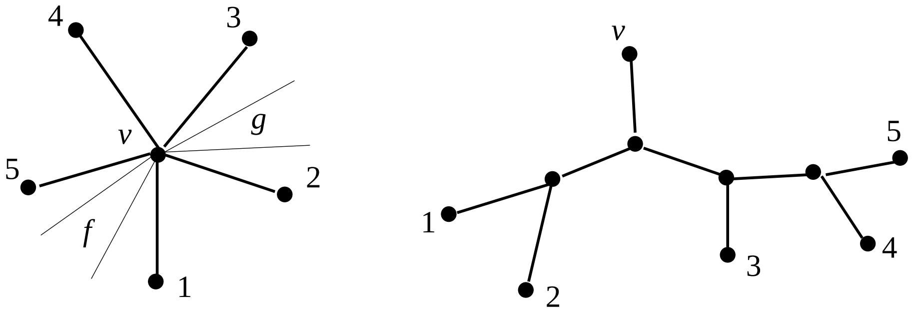
Let T be a medial-axis tree of the given planar graph G. We first construct a local ternary tree C_v for each v. These local trees are later combined into a carving-decomposition of G.

Fix vertex v of G and let k be its degree in T. Let $u_1, \dots, u_k$ be the neigbors of v in T, listed in the counterclockwise order. C_v is a ternary tree with $k+1$ leaves, which are labeled by $v, u_1, \dots, u_k$, constructed as follows.

First suppose v is a leaf of T, i.e., $k = 1$. Then, in this case, C_v is a ternary tree with two leaves and with no internal nodes, one leaf being labeled with v and the other with u_1.

Now suppose $k \leq 2$. Let f, g be a pair of distinct delimiting faces of v that minimizes $\max\{\beta_v(f,g), \beta_v(g,f)\}$. We note that this pair indeed minimizes this quantity over all pairs of faces incident with v, and hence $\max\{\beta_v(f,g), \beta_v(g,f)\} \leq \beta_v$.

Figure 3: Vertex v with its tree edges (left) and its local graph C_v (right)



By renumbering, suppose $E_v(f,g) = \{(v,u_i) \mid 1 \leq i \leq j\}$ and $E_v(g,f) = \{(v,u_i) \mid j < i \leq k\}$ where $1 \leq j < k$. Our local ternary tree C_v is constructed as follows. We start from a ternary tree with three leaves l_1, l_2, l_3, in the counterclockwise order, and one internal node. We label l_1 with v. We extend the tree from l_2 in an arbitrary manner, as long as it remains ternary, until the number of leaves in this extended part is j and label the resulting leaves with $u_1, \ldots, u_j$ in the counterclockwise order. Similarly, we extend the tree from l_3 until the number of leaves in this part is $k - j$ and label the resulting leaves with $u_{j+1}, \ldots, u_k$ in the counterclockwise order. This completes the description of C_v. See Figure 4 for an example.

Having constructed the local tree C_v for each vertex v of G, we now combine them into one tree. Let vertices u, v be adjacent in T. Then, we identify the leaf of C_v labeled u with the leaf of C_u labeled v and then turn the two incident edges into one, removing this identified node. Doing this to all pairs u, v of vertices adjacent in T, we obtain a ternary tree C whose leaves are labeled with the vertices of G: this is our carving-decomposition of G.

It remains to show that the carving-decomposition C thus constructed has width at most $\max\{\beta_G, \delta_G\}$. Consider an arbitrary edge (p,q) of C. If either p or q is a leaf, labeled with $v \in V(G)$, then the width of the separation induced by this edge is the degree of v and therefore at most δ_G. Suppose now that neither p nor q is a leaf. Let v be the vertex of G such that edge (p,q) comes from the local tree C_v. When the choice of v is not unique (which happens when the edge (p,q) is the result of concatenating two edges from two local trees), we choose an arbitrary one of the two possibilities. By abuse of notation, call the corresponding edge in the local tree also (p,q). Let U be the set of labels of the leaves of C_v that lies in the same side of the leaf labeled with v viewed from edge (p,q). Since $U \setminus \{v\}$ is a set of neigbors of v in T that appear contiguously around v in the counterclockwise order, $E_U = \{\{(v,u\} \mid u \in U \setminus \{v\}\}$ can be represented as $E_v(f', g')$ for some delimiting faces f', g' of v. Moreover the construction of C_v guarantees that, this set $E_v(f', g')$ is a superset of either $E_v(f, g)$ or $E_v(g, f)$, where (f, g) is the pair of delimiting faces of v chosen to minimize $\max\{\beta_v(f, g), \beta_v(g, f)\}$ in the construction of C_v. This implies that $\beta_v(g', f')$ is at most $\max\{\beta_v(f, g), \beta_v(g, f)\} \leq \beta_G$. The separation induced by edge (p, q) of C is $\{V(T_v(f', g')), V(G) \setminus V(T_v(f', g')\}$ and by Lemma 3.1, its width is at most $\beta_v(g', f')$. We conclude that the width of each separation in C is at most β_G.

We show that the time required to construct C is linear in the size of G. It suffices to show that the local tree at each v can be constructed in time proportional to the degree of v. The only non-trivial part is the determination of a pair of delimiting faces f, g of v that minimizes $\max\{\beta_v(f, g), \beta_v(g, f)\}$.

For the purpose of analysis, it is easier to consider the problem of finding a pair f, g of any faces incident to v, which is not necessarily a pair of delimiting faces, that minimizes $\max\{\beta_v(f, g), \beta_v(g, f)\}$. These problems are essentially equivalent, because if we find an optimal pair f, g for the latter problem then it can straightforwardly be converted into a pair of delimiting faces f', g' such that $\max\{\beta_v(f, g), \beta_v(g, f)\} = \max\{\beta_v(f', g'), \beta_v(g', f')\}$.

Fix an arbitrary face f incident to v and move face g counterclockwise around v. When g advances one step, one of the following three happens: (1)$\beta_v(f, g)$ increases by one and $\beta_v(g, f)$ decreases by one (when $h(g)$ does not change); (2) $\beta_v(f, g)$ increases by two and $\beta_v(g, f)$ does not change (when $h(g)$ increases); (3) $\beta_v(f, g)$ does not change and $\beta_v(g, f)$ decreases by two (when $h(g)$ decreases). In any case, $\beta_v(f, g) - \beta_v(g, f)$ increases by two. Therefore, among the faces g that minimize $\max\{\beta_v(f, g), \beta_v(g, f)\}$, there is one for which $|\beta_v(f, g) - \beta_v(g, f)| \leq 1$ holds. Call this face g_f. In case of a tie, we choose the one satisfying $\beta_v(f, g_f) > \beta_v(g_f, f)$, for concreteness.

Let f' be the face immediately after f in the counterclockwise order around v. Since $\beta_v(f', g_f) - \beta_v(g_f, f') = \beta_v(f, g_f) - \beta_v(g_f, f) - 2$ and $|\beta_v(f, g_f) - \beta_v(g_f, f)| \leq 1$, we have $\beta_v(f', g_f) < \beta_v(g_f, f')$ and hence $g_{f'}$ cannot lie after f' and strictly before g_f counterclockwise. Therefore, to find a pair f, g_f to minimize $\max\{\beta_v(f, g_f), \beta_v(g_f, f)\}$, we only need two pointers, one for f and the other for g_f, each moving counterclockwise. Neither of these pointers needs to move more than twice around v and therefore our task can be performed in time proportional to the degree of v.

5 Experiments

We implemented the heuristic and compared its performance with the branch-decomposition heuristics for general graphs due to Cook and Seymour [3] and to Hicks [4] on the planar test instances used by these researchers, which are available at [11][1]. These are Delaunay triangulations of point sets taken from TSPLIB[10]. Experimental data of their methods on these instances, as well as their optimum width, are taken from [4].

Tables 2 and 3 in Appendix show the results of the four methods: "Medial-axis" is our heuristic for planar graphs, "diameter" is the diameter method of Hicks, "eigenvector" is the eigenvector method of Cook and Seymour and "hybrid" is a hybrid of the diameter and eigenvector methods due also to Hicks. The number in each instance name indicates the number of vertices of the graph. The CPU used for our experiment is 900MHz Ultra SPARC-III and the CPU for the remaining three methods is 143MHz SPARC1, so we need to keep in mind this difference of speed when we compare the running time. Running times are in seconds, except that "m" means milliseconds. Note that our method is faster than other methods typically by a factor of 10^3 and for some instances by a factor as large as 10^6.

In the width columns, bold face letters indicate that the width is optimum and italic letters indicate that the width is the best among the three methods when none of the three achieved the optimum. Table 1 summarizes the quality of decompositions obtained by each method. For each of the four methods, the number of instances on which the method achieves the width of optimum plus i is shown for each $0 \leq i \leq 4$.

The quality of the decompositions for the first three methods as listed in this table are quite similar, while the eigenvector method gives somewhat inferior results. In view of the great advantage in the running time, our method can be said to exploit the planarity of the instances extremely well.

6 Concluding remarks

Although our main result is formulated for branch decompositions, the underlying theorem on carving decompositions (Theorem 2.2) is equally important in its

[1]Instances pr76, pcb442, and vm1084 are excluded, because the point set files for these instances are missing from the archive at [11] and the files that describe edges do not produce planar graphs when combined with the standard TSPLIB point set files.

Table 1: Quality of the decompositions: summary

	Medial-axis	Diameter	Hybrid	Eigenvector
opt	33	34	35	23
opt + 1	16	15	15	12
opt + 2	7	3	7	4
opt + 3	2	4	0	3
opt + 4	1	0	1	6
greater	0	3	1	11

own right, because for some problems the carving-decomposition is more suitable than the branch-decomposition as the framework for dynamic programming. In fact, the carving-decomposition heuristic given by Theorem 2.2 is one of the core components in our TSP solver for sparse planar graphs [8], which has recently discovered new best tours for two unsolved TSPLIB instances brd14051 and d18512 [9]. This solver takes the approach similar to that of Cook and Seymour [3] who use branch-decomposition to solve TSP on small-width graphs that arise from the union of near-optimal tours. The advantage of our solver comes in large part from the heuristic described in this paper which enables the quick recognition of small-width graphs and gives high quality decompositions.

We envision applications of our fast branch- and carving-decomposition methods to other areas, such as various optimization problems on surface meshes which occur in graphics and geometric modeling context. The approach would consist of repeated improvements, in each step of which we take a submesh, whose width as a planar graph is small enough, and compute an optimal local improvement on the submesh via dynamic programming. The quick recognition of small-width submeshes is crucial in such applications as well and the heuristic introduced in this paper would be an indispensable tool.

Although the effectiveness of our heuristic has been verified by some test instances and through an application, there is no theoretical guarantee on the quality of the decomposition it produces. In fact, for any given $r > 0$, it is easy to construct an example for which our heuristic produces a decomposition with width that is more than r times greater than the optimum width. A fast algorithm with some guarantee on the quality of decomposition would be of great interest.

Acknowledgment

This work was done while the author was visiting Max Planck Institute for Computer Science at Saarbrücken. He thanks its algorithm group for the hospitality and for the stimulating environment.

References

[1] S. Arnborg and J. Lagergren and D. Seese, Easy problems for tree-decomposable graphs, Journal of Algorithms, 12, 308-340, 1991

[2] H. Bodlaender, A tourist guide through treewidth, Acta Cybernetica, 11, 1-21, 1993.

[3] W. Cook and P.D. Seymour, Tour merging via branch-decomposition, to appear in INFORMS Jounal on Computing, 2003

[4] I.V. Hicks, Branch Decompositions and their applications, Phd thesis, Rice University, April, 2000.

[5] N. Robertson and P.D. Seymour, Graph minors II. Algorithmic aspects of tree width, Journal of Algorithms, 7, 309-322, 1986

[6] N. Robertson and P.D. Seymour, Graph minors X. Obstructions to tree-decomposition, Journal of Combinatorial Theory, Series B, 153-190, 1991

[7] P.D. Seymour and R. Thomas, Call routing and the ratcatcher. Combinatorica, 14(2), 217-241, 1994

[8] H. Tamaki, Solving TSP on sparse planar graphs, in preparation.

[9] TSP home page by D. Applegate, R. Bixby, W. Cook and V. Chvátal: http://www.math.princeton.edu/tsp/

[10] TSPLIB by Gerd Reinelt: http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/

[11] Planar graph test data by I.V.Hicks: http://ie.tamu.edu/People/faculty/Hicks/data.html

Appendix

Table 2: Experimental results: 900MHz Ultra SPARC-III for Medial-axis and 143MHz SPARC1 for the other methods (data taken from [4]); times are in seconds, "m" means milliseconds.

instance	Medial-axis		Diameter		Hybrid		Eigenvector		optimal
	width	time	width	time	width	time	width	time	width
a280	**13**	23m	**13**	10	**13**	52	14	11	13
bier127	**14**	15m	**14**	5	**14**	7	**14**	5	14
ch130	**10**	5m	**10**	2	**10**	11	11	4	10
ch150	**12**	2m	**12**	5	**12**	15	**12**	5	12
d1291	*30*	9m	*30*	6956	*30*	1881	38	1399	29
d1655	*30*	11m	32	15318	31	9031	45	1714	29
d198	**12**	1m	**12**	12	**12**	13	14	11	12
d2103	**29**	17m	**29**	12025	**29**	15336	35	2332	29
d493	22	4m	**20**	385	21	255	24	79	20
d657	23	5m	**22**	442	**22**	356	26	135	22
eil101	**10**	0m	**10**	1	**10**	4	11	2	10
eil51	**8**	0m	**8**	0.0	**8**	0.0	**8**	0.0	8
eil76	**10**	0m	**10**	0.0	**10**	4	11	2	10
fl1400	17	12m	**13**	8250	14	5406	14	1143	13
fl1577	**16**	17m	19	11157	**16**	87	19	554	16
fl417	11	3m	**9**	56	**9**	41	**9**	45	9
gil262	**15**	1m	**15**	29	**15**	41	16	21	15
kroA100	10	0m	**9**	1	**9**	2	**9**	2	9
kroA150	**11**	1m	**11**	6	**11**	8	**11**	5	11
kroA200	**11**	2m	**11**	11	**11**	12	**11**	12	11
kroB100	10	0m	**9**	1	**9**	1	**9**	2	9
kroB150	**10**	1m	**10**	6	**10**	10	**10**	6	10
kroB200	**12**	1m	**12**	10	**12**	11	13	11	12
kroC100	**9**	0m	**9**	1	**9**	2	**9**	2	9
kroE100	9	0m	**8**	1	**8**	2	**8**	2	8
lin105	9	1m	**8**	1	**8**	1	9	2	8

Table 3: Experimental results: continued

instance	Medial-axis		Diameter		Hybrid		Eigenvector		optimal
	width	time	width	time	width	time	width	time	width
nrw1379	**31**	9m	**31**	8907	**31**	8456	38	1216	31
p654	12	6m	12	170	11	365	**10**	143	10
pcb1173	**25**	90m	27	3511	27	2744	31	727	25
pr1002	24	7m	24	1053	*23*	1864	24	450	21
pr107	7	0m	7	1	7	3	**6**	2	6
pr124	10	1m	**8**	2	**8**	4	**8**	1	8
pr136	**10**	0m	**10**	4	**10**	7	**10**	3	10
pr144	**9**	1m	**9**	3	**9**	5	**9**	3	9
pr152	11	1m	**8**	3	**8**	8	**8**	4	8
pr226	**7**	1m	**7**	7	**7**	20	**7**	5	7
pr2392	*30*	24m	34	47605	33	48728	34	4086	29
pr264	**13**	1m	14	20	14	30	**13**	21	13
pr299	**11**	2m	**11**	14	**11**	24	**11**	16	11
pr439	**16**	3m	17	171	17	167	17	59	16
rat195	**12**	1m	**12**	9	**12**	11	**12**	9	12
rat575	*18*	3m	*18*	588	*18*	396	19	105	17
rat783	*21*	6m	*21*	1065	*21*	1416	23	212	20
rat99	**9**	0m	**9**	1	**9**	1	**9**	2	9
rd100	**10**	1m	11	1	11	4	**10**	2	10
rd400	*18*	2m	*18*	79	*18*	92	19	50	17
rl1304	*20*	10m	*20*	15205	21	2723	21	906	19
rl1323	*23*	9m	*23*	4686	*23*	4131	26	900	22
rl1889	24	19m	25	13130	*23*	11707	26	2487	22
tsp225	**12**	1m	**12**	20	**12**	15	13	11	12
u1060	21	8m	26	2582	**20**	2176	37	573	20
u1432	**32**	11m	33	11880	34	5971	40	1238	32
u159	12	1m	**10**	6	**10**	10	11	4	10
u1817	**28**	16m	29	14081	29	10540	34	2005	28
u2152	**31**	18m	36	36887	37	22979	39	2698	31
u2319	**44**	117m	46	66077	46	23510	72	3264	44
u574	18	4m	18	316	**17**	239	18	100	17
u724	**18**	20m	19	2178	19	547	22	177	18
vm1748	**22**	13m	23	11982	24	10495	26	1164	22

Below you find a list of the most recent technical reports of the Max-Planck-Institut für Informatik. They are available by anonymous ftp from `ftp.mpi-sb.mpg.de` under the directory `pub/papers/reports`. Most of the reports are also accessible via WWW using the URL `http://www.mpi-sb.mpg.de`. If you have any questions concerning ftp or WWW access, please contact `reports@mpi-sb.mpg.de`. Paper copies (which are not necessarily free of charge) can be ordered either by regular mail or by e-mail at the address below.

Max-Planck-Institut für Informatik
Library
attn. Anja Becker
Stuhlsatzenhausweg 85
66123 Saarbrücken
GERMANY
e-mail: `library@mpi-sb.mpg.de`

MPI-I-2003-NWG2-002	F. Eisenbrand	Fast integer programming in fixed dimension
MPI-I-2003-NWG2-001	L.S. Chandran, C.R. Subramanian	Girth and Treewidth
MPI-I-2003-4-001	M. Tarini, H.P.R. Lensch, M. Goesele, H. Seidel	3-D Aqusition of Mirrowing Objects
MPI-I-2003-2-001	P. Maier	Compositional Circular Assume-Guarantee Rules Cannot Be Sound And Complete
MPI-I-2003-1-010	H. Tamaki	Soon to be published
MPI-I-2003-1-009	B. Csaba	On the Bollobás – Eldridge conjecture for bipartite graphs
MPI-I-2003-1-008	P. Sanders	Soon to be published
MPI-I-2003-1-007	H. Tamaki	Alternating cycles contribution: a strategy of tour-merging for the traveling salesman problem
MPI-I-2003-1-006	H. Tamaki, M. Dietzfelbinger	On the probability of Rendezvous in Graph
MPI-I-2003-1-005	M. Dietzfelbinger, P. Woelfel	Almost Random Graphs with Simple Hash Functions
MPI-I-2003-1-004	E. Althaus, T. Polzin, S.V. Daneshmand	Improving Linear Programming Approaches for the Steiner Tree Problem
MPI-I-2003-1-003	R. Beier, B. V÷*cking*	Random Knapsack in Expected Polynomial Time
MPI-I-2003-1-002	P. Krysta, P. Sanders, B. V÷*cking*	Scheduling and Traffic Allocation for Tasks with Bounded Splittability
MPI-I-2003-1-001	P. Sanders, R. Dementiev	Asynchronous Parallel Disk Sorting
MPI-I-2002-4-002	F. Drago, W. Martens, K. Myszkowski, H. Seidel	Perceptual Evaluation of Tone Mapping Operators with Regard to Similarity and Preference
MPI-I-2002-4-001	M. Goesele, J. Kautz, J. Lang, H.P.A. Lensch, H. Seidel	Tutorial Notes ACM SM 02 A Framework for the Acquisition, Processing and Interactive Display of High Quality 3D Models
MPI-I-2002-2-008	W. Charatonik, J. Talbot	Atomic Set Constraints with Projection
MPI-I-2002-2-007	W. Charatonik, H. Ganzinger	Symposium on the Effectiveness of Logic in Computer Science in Honour of Moshe Vardi
MPI-I-2002-1-008	P. Sanders, J.L. Trõff	The Factor Algorithm for All-to-all Communication on Clusters of SMP Nodes
MPI-I-2002-1-005	M. Hoefer	Performance of heuristic and approximation algorithms for the uncapacitated facility location problem
MPI-I-2002-1-004	S. Hert, T. Polzin, L. Kettner, G. Schõfer	Exp Lab A Tool Set for Computational Experiments

MPI-I-2002-1-003	I. Katriel, P. Sanders, J.L. Trõff	A Practical Minimum Scanning Tree Algorithm Using the Cycle Property
MPI-I-2002-1-002	F. Grandoni	Incrementally maintaining the number of l-cliques
MPI-I-2002-1-001	T. Polzin, S. Vahdati	Using (sub)graphs of small width for solving the Steiner problem
MPI-I-2001-4-005	H.P.A. Lensch, M. Goesele, H. Seidel	A Framework for the Acquisition, Processing and Interactive Display of High Quality 3D Models
MPI-I-2001-4-004	S.W. Choi, H. Seidel	Linear One-sided Stability of MAT for Weakly Injective Domain
MPI-I-2001-4-003	K. Daubert, W. Heidrich, J. Kautz, J. Dischler, H. Seidel	Efficient Light Transport Using Precomputed Visibility
MPI-I-2001-4-002	H.P.A. Lensch, J. Kautz, M. Goesele, H. Seidel	A Framework for the Acquisition, Processing, Transmission, and Interactive Display of High Quality 3D Models on the Web
MPI-I-2001-4-001	H.P.A. Lensch, J. Kautz, M. Goesele, W. Heidrich, H. Seidel	Image-Based Reconstruction of Spatially Varying Materials
MPI-I-2001-2-006	H. Nivelle, S. Schulz	Proceeding of the Second International Workshop of the Implementation of Logics
MPI-I-2001-2-005	V. Sofronie-Stokkermans	Resolution-based decision procedures for the universal theory of some classes of distributive lattices with operators
MPI-I-2001-2-004	H. de Nivelle	Translation of Resolution Proofs into Higher Order Natural Deduction using Type Theory
MPI-I-2001-2-003	S. Vorobyov	Experiments with Iterative Improvement Algorithms on Completely Unimodel Hypercubes
MPI-I-2001-2-002	P. Maier	A Set-Theoretic Framework for Assume-Guarantee Reasoning
MPI-I-2001-2-001	U. Waldmann	Superposition and Chaining for Totally Ordered Divisible Abelian Groups
MPI-I-2001-1-007	T. Polzin, S. Vahdati	Extending Reduction Techniques for the Steiner Tree Problem: A Combination of Alternative-and Bound-Based Approaches
MPI-I-2001-1-006	T. Polzin, S. Vahdati	Partitioning Techniques for the Steiner Problem
MPI-I-2001-1-005	T. Polzin, S. Vahdati	On Steiner Trees and Minimum Spanning Trees in Hypergraphs
MPI-I-2001-1-004	S. Hert, M. Hoffmann, L. Kettner, S. Pion, M. Seel	An Adaptable and Extensible Geometry Kernel
MPI-I-2001-1-003	M. Seel	Implementation of Planar Nef Polyhedra
MPI-I-2001-1-002	U. Meyer	Directed Single-Source Shortest-Paths in Linear Average-Case Time
MPI-I-2001-1-001	P. Krysta	Approximating Minimum Size 1,2-Connected Networks
MPI-I-2000-4-003	S.W. Choi, H. Seidel	Hyperbolic Hausdorff Distance for Medial Axis Transform
MPI-I-2000-4-002	L.P. Kobbelt, S. Bischoff, K. Kõhler, R. Schneider, M. Botsch, C. R÷$ssl, J. Vorsatz$	Geometric Modeling Based on Polygonal Meshes
MPI-I-2000-4-001	J. Kautz, W. Heidrich, K. Daubert	Bump Map Shadows for OpenGL Rendering
MPI-I-2000-2-001	F. Eisenbrand	Short Vectors of Planar Lattices Via Continued Fractions
MPI-I-2000-1-005	M. Seel, K. Mehlhorn	Infimaximal Frames: A Technique for Making Lines Look Like Segments
MPI-I-2000-1-004	K. Mehlhorn, S. Schirra	Generalized and improved constructive separation bound for real algebraic expressions
MPI-I-2000-1-003	P. Fatourou	Low-Contention Depth-First Scheduling of Parallel Computations with Synchronization Variables
MPI-I-2000-1-002	R. Beier, J. Sibeyn	A Powerful Heuristic for Telephone Gossiping
MPI-I-2000-1-001	A branch and cut algorithm for the optimal solution of the side-chain placement problem	

MPI-I-1999-4-001	J. Haber, H. Seidel	A Framework for Evaluating the Quality of Lossy Image Compression
MPI-I-1999-3-005	T.A. Henzinger, J. Raskin, P. Schobbens	Axioms for Real-Time Logics
MPI-I-1999-3-004	J. Raskin, P. Schobbens	Proving a conjecture of Andreka on temporal logic
MPI-I-1999-3-003	T.A. Henzinger, J. Raskin, P. Schobbens	Fully Decidable Logics, Automata and Classical Theories for Defining Regular Real-Time Languages