INFORMS Journal on Computing
Vol. 17, No. 4, Fall 2005, pp. 413–421
ISSN 1091-9856 | EISSN 1526-5528 | 05 | 1704 | 0413

DOI 10.1287/ijoc.1040.0074

Planar Branch Decompositions II: The Cycle Method

Illya V. Hicks
Department of Industrial Engineering, Texas A & M University, College Station, Texas 77843-3131, USA,
ivhicks@tamu.edu

This is the second of two papers dealing with the relationship of branchwidth and planar graphs. Branchwidth and branch decompositions, introduced by Robertson and Seymour, have been shown to be beneficial for both proving theoretical results on graphs and solving NP-hard problems modeled on graphs. The first practical implementation of an algorithm of Seymour and Thomas for computing optimal branch decompositions of planar hypergraphs is presented. This algorithm encompasses another algorithm of Seymour and Thomas for computing the branchwidth of any planar hypergraph, whose implementation is discussed in the first paper. The implementation also includes the addition of a heuristic to decrease the run times of the algorithm. This method, called the cycle method, is an improvement on the algorithm by using a "divide-and-conquer" approach.

Key words: planar graph; branchwidth; branch decomposition; carvingwidth
History: Accepted by Jan Karel Lenstra, former Area Editor for Design and Analysis of Algorithms; received December 2002; revised August 2003, January 2004; accepted February 2004.

1. Introduction

The notion of branch decompositions and its related connectivity invariant for graphs, branchwidth, were created by Robertson and Seymour (1985) to assist in the proof of the graph minors theorem; however, branch decompositions have also been used in other theoretical results and to produce algorithms to solve NP-hard problems modeled on graphs with bounded branchwidth in polynomial-time (Arnborg et al. 1991). For example, branchwidth and branch decompositions have been shown useful in producing matroid analogs of the graph minors theorem and shorter proofs of the graph minors theorem related to graphs with bounded branchwidth/treewidth (Geelen et al. 2002). In addition, practical branch decomposition-based algorithms have been proposed for the traveling salesman problem (TSP) (Cook and Seymour 2003) and general minor containment (Hicks 2004a). This paper is the second of two papers based on the relationship between branchwidth and planar graphs. Thus, a more detailed discussion of these topics and other related background can be found in the related paper (Hicks 2005).

Seymour and Thomas (1994) proposed a polynomial-time algorithm to compute the branchwidth for planar graphs, called the *ratcatcher* method. Given a planar graph and an integer k, the ratcatcher method tests whether the branchwidth of the graph is at least k by finding a structure in the graph that prohibits finding a branch decomposition of the graph with width at most $k-1$. Thus, the ratcatcher method computes only the branchwidth of a planar graph and not an optimal branch decomposition. A detailed discussion of the implementation of the ratcatcher method is presented in the related paper (Hicks 2005). Seymour and Thomas (1994) also proposed a polynomial-time algorithm, using the ratcatcher method, to compute optimal branch decompositions for planar graphs, called the *edge-contraction* method. The edge-contraction method finds optimal branch decompositions by finding bipartitions of the edge sets for multiple planar hypergraphs. Unfortunately, the edge-contraction method only finds a pair of hyperedges as one side of the bipartitions; the method ignores the "divide-and-conquer" power of finding bipartitions where the cardinalities of the sets are close to each other. This paper offers a practical implementation of the edge-contraction method by adding a heuristic. This method, called the *cycle method*, is an improvement on the edge-contraction method by using a "divide-and-conquer" approach.

Foundational definitions are given in §§2.1 and 2.2. Section 3 describes the tree-building techniques needed for both the edge-contraction method and the cycle method. Section 4 offers a discussion of a property that guarantees the correctness of the algorithms. Sections 5 and 6 are reserved for the details of the algorithms and §§7 and 8 contain computational results and conclusions, respectively.

2. Preliminaries

2.1. Branch Decompositions

Throughout this paper, a graph is only considered to be undirected and may have loops and multiple edges unless stated otherwise. For a graph G, one can derive an optimal branch decomposition for G from the optimal branch decompositions of G's connected components. In addition, for a connected graph H, one can derive an optimal branch decomposition for H from the optimal branch decompositions of the biconnected components of H. Thus, we may assume all graphs to be biconnected.

A *hypergraph* H consists of a finite set $V(H)$ of nodes, a finite set $E(H)$ of edges, and an incidence relation between them that is not restricted to two ends for each edge. A graph is planar if it can be drawn on the plane or sphere such that no edges cross. Let H be a hypergraph; then $I(H)$, the *incidence graph* of H, is the simple bipartite graph with node set $V(H) \cup E(H)$ such that $v \in V(H)$ is adjacent in $I(H)$ to $e \in E(H)$ if and only if v is an end of e in H. Seymour and Thomas (1994) define a hypergraph H as being planar if and only if $I(H)$ is planar. In addition, we will denote G^* as a planar dual of an embedding of the graph G in the sphere or plane.

Let G be a graph with node set $V(G)$ and edge set $E(G)$. Let T be a tree having $|E(G)|$ leaves in which every nonleaf node has degree three. Let ν be a bijection between the edges of G and the leaves of T. The pair (T, ν) is called a *branch decomposition* of G. Notice that removing an edge e of T partitions the edges of G into two subsets A_e and B_e. The *middle set* of e, denoted mid(e), is the set $V(A_e) \cap V(B_e)$. The *width* of a branch decomposition (T, ν) is the maximum cardinality of the middle sets over all edges in T. The *branchwidth* of G, denoted $\beta(G)$, is the minimum width over all branch decompositions of G. A branch decomposition of G is *optimal* if its width is equal to the branchwidth of G. If every nonleaf node of T has a degree of at least three then the pair (T, ν) is called a *partial branch decomposition*. Figure 2, providing some of the middle sets, illustrates an optimal branch decomposition of the graph of Figure 1. Other examples of branchwidth for different graph classes can be found in the related paper (Hicks 2005) or Hicks (2000).

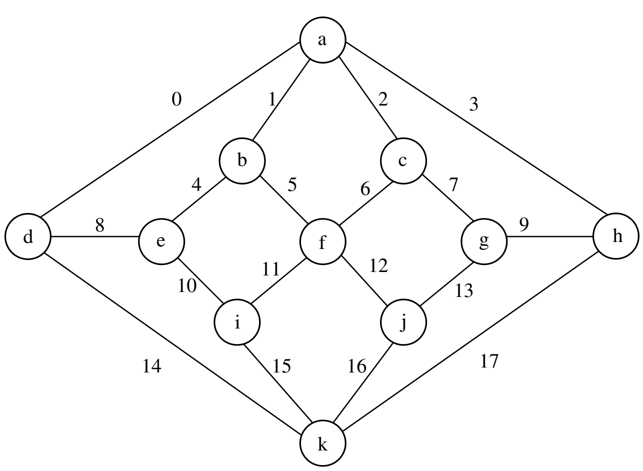

Figure 1 Example Planar Graph

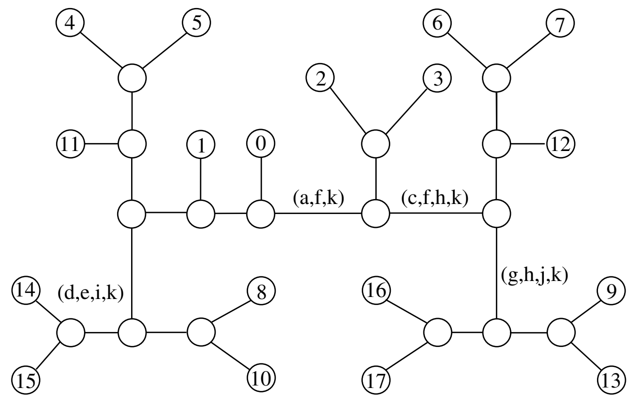

Figure 2 Optimal Branch Decomposition for Figure 1

Given a nonplanar graph G, one could use the edge-contraction method or the cycle method to find a bound on the branchwidth of G using the following lemma. A graph H is a *minor* of a graph G if H can be obtained from a subgraph of G by contracting edges.

Lemma 1. *Let G be a biconnected graph and let H denote a subgraph of G. Then $\beta(H) \le \beta(G) \le \beta(H) + 2|E(G\backslash H)|$, and if H is biconnected, then $\beta(H) \le \beta(G) \le \beta(H) + |E(G\backslash H)|$.*

Proof. The lower bound is a result of Robertson and Seymour (1991) relating the branchwidth of a graph with any minor of that graph. For the upper bound, let (T, ν) be the optimal branch decomposition for H. Let $e \in E(G\backslash H)$. Suppose e has ends u and v. There are three cases to consider: There exists an edge of T that has u and v in its middle set; there is at least one edge of T that contains at least u or v, say v, in its middle set; and neither u nor v is contained in the middle set of any edge of T. For the first case, let f denote the edge of T. Subdivide f in T to create T_o where w is the new node created. In addition, add the edge wx to T_o where x is a new leaf node. Then define ν_o such that $\nu_o(t) = \nu(t)$ for all $t \in E(H)$ and $\nu_o(e) = x$. Thus, (T_o, ν_o) is a branch decomposition of $H \cup \{e\}$ with the equivalent width of (T, ν). For the second case, let f denote the edge of T that contains v in its middle set. Perform the same procedure to create (T_o, ν_o) and one can see that the width of (T_o, ν_o) is one greater than the width of (T, ν). For the third case, let t be a leaf of T such that $\nu^{-1}(t)$ is an edge incident to v. We know t exists since G is biconnected and H is spanning by definition. Let f denote the edge of T

incident to t. Perform the same procedure to create (T_o, ν_o) and one can see that the width of (T_o, ν_o) is two greater than the width of (T, ν). Continue the process until (T_o, ν_o) is a branch decomposition of G. Furthermore, one would not have to consider case 3 if H were biconnected. □

Simply find a maximal planar subgraph of G, polynomial-time solvable (see Liebers 2001 for a survey on the subject of finding planar subgraphs), and perform the procedure in the proof of Lemma 1 to find a near-optimal branch decomposition. Notice that the upper bound given in Lemma 1 can be poor if $|E(G\backslash H)|$ is large. Another direction to use the edge-contraction method or the cycle method to find near-optimal branch decompositions for nonplanar graphs is to use the methods in conjunction with one of the heuristics developed by Cook and Seymour (2003) or Hicks (2000, 2002) to obtain another heuristic for computing near-optimal branch decompositions for nonplanar graphs. This hybrid heuristic would give optimal branch decompositions for planar graphs and would offer near-optimal branch decompositions with widths at most the widths of the near-optimal branch decompositions offered by the stand-alone heuristic. Other background material and discussions pertaining to constructing near-optimal branch decompositions (including the use of tree decompositions) can be found in the related paper (Hicks 2005).

2.2. Carving Decompositions

Let G be a graph with node set $V(G)$ and edge set $E(G)$. Let T be a tree having $|V(G)|$ leaves in which every nonleaf node has a degree of three. Let μ be a bijection between the nodes of G and the leaves of T. The pair (T, μ) is called a *carving decomposition* of G. Notice that removing an edge e of T partitions the nodes of G into two subsets A_e and B_e. The *cut set* of e is the set of edges that are incident to nodes in A_e and to nodes in B_e. The *width* of a carving decomposition (T, μ) is the maximum cardinality of the cut sets for all edges in T. The *carvingwidth* for G, $\kappa(G)$, is the minimum width over all carving decompositions of G. A carving decomposition is also known as a *minimum-congestion routing tree*; see Alvarez et al. (2000) for a link between carvingwidth and network design.

Let G be a planar graph and represent the planar embedding of the graph on the sphere. For every node v of G, the edges incident to v can be ordered in a clockwise or counterclockwise order. This ordering of edges incident to v is the *cyclic order* of v. Let $M(G)$ be a graph with the node set $E(G)$ and let the edges of $M(G)$ be the union of cycles C_v for all $v \in V(G)$ where C_v is the cycle through the nodes of $M(G)$ that correspond to the edges incident with v according to v's cyclic order. $M(G)$ is called a *medial graph* of G. The medial graph of Figure 1 is given in Figure 3. Notice that every connected planar hypergraph G with $E(G) \neq \varnothing$ has a planar medial graph. In addition, notice that there is a bijection between the regions of $M(G)$ and the nodes and regions of G. Using this relationship, Hicks (2000) proved that if a planar graph and its dual are both loopless then they have the same branchwidth. For the relationship between branchwidth and carvingwidth, Seymour and Thomas (1994) proved:

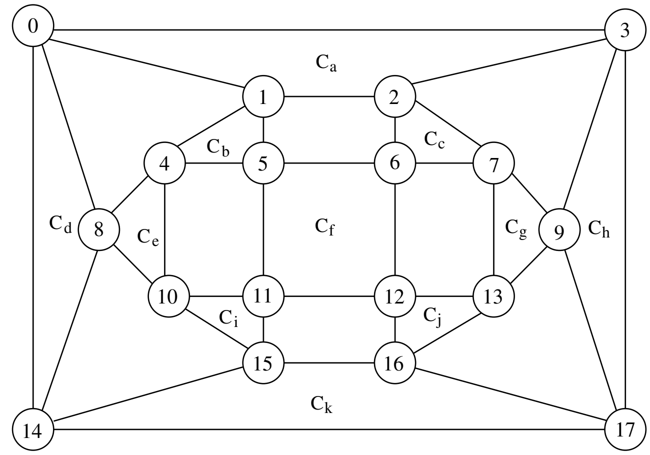

Figure 3 **Medial Graph of Figure 1**

Theorem 1 (Seymour and Thomas 1994). *Let G be a connected planar graph with $|E(G)| \geq 2$, and let $M(G)$ be a medial graph of G. Then the branchwidth of G is half the carvingwidth of $M(G)$.*

Therefore, computing the carvingwidth of $M(G)$ gives us the branchwidth of G. Also, having a carving decomposition of $M(G)$, (T, μ), gives us a branch decomposition of G, (T, ν), such that the width of (T, ν) is exactly half the width of (T, μ). Thus, the edge-contraction method actually computes an optimal carving decomposition of a planar graph. A more detailed discussion on the relationship between carvingwidth and branchwidth can be found in the related paper (Hicks 2005).

3. Tree Building

How does one build optimal branch decompositions of planar graphs using the edge-contraction method or the cycle method? First, we compute the branchwidth of the input graph using the ratcatcher method (Seymour and Thomas 1994). We will not discuss the details of the ratcatcher method, but a practical implementation of the ratcatcher method is presented in the related paper (Hicks 2005). For this paper, we may assume that the branchwidth of any planar graph is available. Next, we use the edge-contraction method or the cycle method to build an optimal branch decomposition of the graph. Before we can discuss the edge-contraction method and the cycle method fully, we must define a few more terms and discuss how to build a branch decomposition from a partial branch decomposition.

A *separation* of a hypergraph G is a pair (G_1, G_2) of subhypergraphs of G with $G_1 \cup G_2 = (V(G_1) \cup V(G_2), E(G_1) \cup E(G_2)) = G$ and $E(G_1) \cap E(G_2) = \emptyset$, and the *order* of this separation is defined as $|V(G_1) \cap V(G_2)|$. Without loss of generality, we only use separations (G_1, G_2) such that $E(G_1)$ and $E(G_2)$ are nonempty. Separations will be vital to our construction of the branch decomposition because finding separations will help partial branch decompositions grow into optimal branch decompositions.

Let G represent our input planar graph and let (T_1, ν) be a partial branch decomposition of G. Let v be a nonleaf node of T_1 with a degree greater than three and denote D_v as the set of edges incident with v. For a set $S \subseteq V(G)$, let $he(S)$ denote a hyperedge where the ends of the hyperedge are the elements in S. Define H^v as the hypergraph constructed from the union of hyperedges $he(\text{mid}(e))$ for all $e \in D_v$. Therefore if T_1 was a star, then H^v would correspond to G because G is assumed to be biconnected. Let (X, Y) be a separation of H^v. Using (X, Y), one can create a new tree T_2 such that v is replaced by nodes x and y and edge xy where x would be incident to the edges that correspond to $E(X)$ and y would be incident to the edges that correspond to $E(Y)$. This procedure is called a *one split*. The middle set for the edge xy would be $V(X) \cap V(Y)$ and Figure 4 offers an illustration of a one split.

Let G, (T_1, ν), and v be defined as in the previous paragraph. Let e be an edge incident with v and let $he(e)$ denote the hyperedge of H^v that corresponds to e. Let (X, Y) be a separation of the hypergraph $H^v \backslash \{he(e)\}$. Without loss of generality, assume that the cardinality of $E(X)$ is at most the cardinality of $E(Y)$. If the cardinality of $E(X)$ is greater than one, one can create a new tree T_2 by adding new nodes x and y and edges vx and vy to T_1 with x incident to the edges corresponding to $E(X)$ and y incident to the edges corresponding to $E(Y)$. Otherwise, create T_2 by creating a new node y and edge vy with y incident to the edges corresponding to $E(Y)$. The middle sets of the new edges in either case would be:

$$\text{mid}(vx) = (V(Y) \cup \text{mid}(e)) \cap V(X); \quad (1)$$

$$\text{mid}(vy) = (V(X) \cup \text{mid}(e)) \cap V(Y). \quad (2)$$

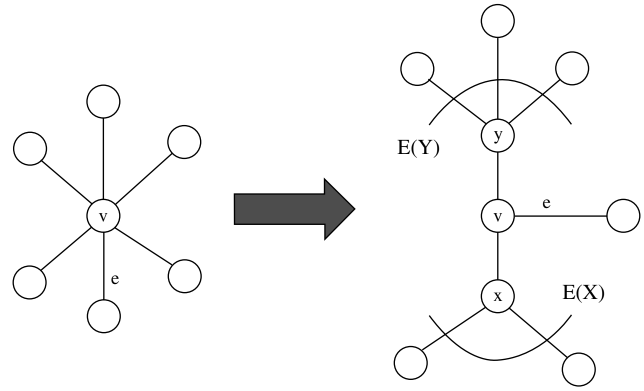

Figure 5 **A Two Split if** $|X| > 1$

This procedure is called a *two split*. A two split when $|E(X)| > 1$ is given in Figure 5, while Figure 6 gives an example of a two split when $|E(X)| = 1$. Notice that a two split when $|E(X)| = 1$ is equivalent to an one split when $|E(X)| = 2$. This fact is used when finding greedy separations, which are discussed in §4. Otherwise, the two procedures do not yield the same results.

To build a branch decomposition, we would start with a partial branch decomposition whose tree is a star, and then use successive one splits or two splits or both to achieve a branch decomposition. The tree-building aspect of using only one splits is equivalent to the tree-building aspect developed by Cook and Seymour (2003), and the tree-building aspect of using only two splits is equivalent to the tree-building aspect developed by Robertson and Seymour (1995). The edge-contraction method uses only one splits,

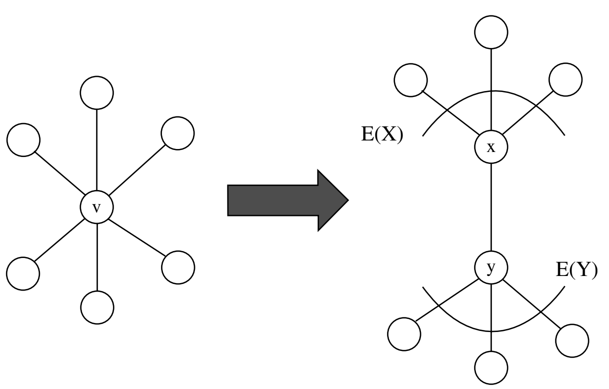

Figure 4 **A One Split**

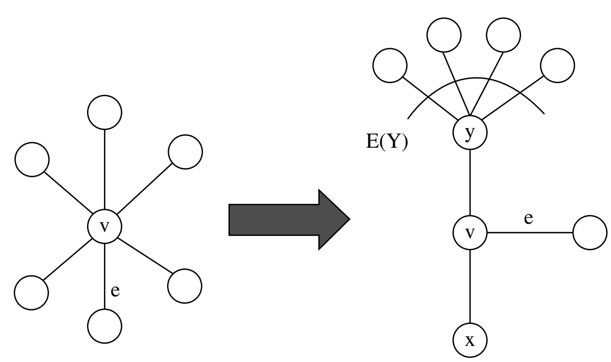

Figure 6 **A Two Split if** $|X| = 1$

while the cycle method can use both one and two splits. Notice that two splits offer two one splits for the price of one. This fact is one of the main reasons why the cycle method is inherently faster that the edge-contraction method.

4. Extendible Partial Branch Decompositions

A partial branch decomposition (T, ν) of a graph G is called *extendible* if $\beta(H^v) \leq \beta(G)$ for every nonleaf node $v \in V(T)$. This follows from the fact that if every H^v had branchwidth at most some number k, then one could use the optimal branch decompositions of the hypergraphs to build a branch decomposition of G whose width is at most k. This definition of extendible is synonymous with that given by Cook and Seymour (2003). Even though a partial branch decomposition whose tree is a star is extendible, there is not a way to check whether a partial branch decomposition is extendible for general graphs. In contrast, we have the ratcatcher method to check whether a partial branch decomposition of a planar graph is extendible. Thus, in both the edge-contraction and cycle method, we use the ratcatcher method to verify that the partial branch decompositions created are still extendible. This is the key proof that both methods find optimal branch decompositions for planar graphs.

A separation is called *greedy* if the next partial branch decomposition created by the use of the separation in conjunction with a one or two split is extendible if the previous partial branch decomposition was extendible. Cook and Seymour (2003) called these type of separations *safe*. In particular, Cook and Seymour describe three types of safe separations, but for the computational code described in this paper, only the first, and the more general type called *push* was used. For a hypergraph H, denote $H[F]$ as the subhypergraph of H induced by F, a subset of nodes or edges. The push separation is described in the following lemma.

Lemma 2 (Cook and Seymour 2003). *Let G be a graph with a partial branch decomposition (T, ν). Let $v \in V(T)$ have a degree greater than three and let $D_v \subseteq E(T)$ be the set of edges incident with v. Also, let H^v be the corresponding hypergraph for v. Suppose there exist $e_1, e_2 \in E(T)$ incident with v such that $|(\text{mid}(e_1) \cup \text{mid}(e_2)) \cap \bigcup\{\text{mid}(f) \mid f \in D_v \backslash \{e_1, e_2\}\}| \leq \max(|\text{mid}(e_1)|, |\text{mid}(e_2)|)$. Let $h_{e_1}, h_{e_2} \in E(H^v)$ be the corresponding hyperedges for e_1 and e_2, respectively. Then the resulting partial branch decomposition after taking a one split using the separation $(H^v[\{h_{e_1}, h_{e_2}\}], H^v[E(H^v)\backslash\{h_{e_1}, h_{e_2}\}])$ is extendible if T was extendible.*

Greedy separations were also used by Hicks (2002) to produce near-optimal branch decompositions for general graphs. As seen in §7, the use of greedy separations will be beneficial to the overall speedup of both the edge-contraction and the cycle method.

5. The Edge-Contraction Method

Before we can discuss the cycle method, we must define the edge-contraction method. Suppose that (T, ν) is a partial branch decomposition for the input planar graph G. The method proceeds as follows:

While $\exists$ a node v of T with degree greater than three
  Compute $M(H^v)$
  For each $st \in E(M(H^v))$
    If $M(H^v)\backslash\{s, t\}$ is connected
    then Let M_o be obtained from $M(H^v)$ by contracting all edges between s and t
      If M_o has carvingwidth $\leq 2 * \beta(G)$
      then use the separation $(H^v[\{s, t\}], H^v[E(H^v)\backslash\{s, t\}])$ for a one split to create the new partial branch decomposition T_o. Replace T by T_o and break.
    end
  end
end

Since G is biconnected, $M(G)$ is loopless and biconnected. We verify that $M(H^v)\backslash\{s, t\}$ is connected to ensure that M_o is loopless and biconnected because the ratcatcher method assumes that all medial graphs created are loopless and biconnected (Seymour and Thomas 1994). Also, M_o is the medial graph of the hypergraph H^y of the new node y in the next partial branch decomposition. The carvingwidth of M_o is checked to keep the partial branch decomposition extendible.

For the complexity of the edge-contraction method, given some integer $k \geq 1$, Seymour and Thomas (1994) state that the complexity to test whether a planar graph has branchwidth $\geq k$ using the ratcatcher method is $O(e^2)$ where $e = |E(G)|$. The original complexity result was on the nodes and edges of the initial medial graph but since $|V(M(G)) + E(M(G))| = 3e$, the previous result is valid. Since the branchwidth of a graph is bounded by the nodes of the graph (Robertson and Seymour 1991), the complexity to find the

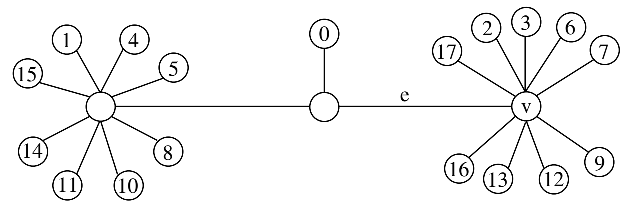

Figure 7 Example Partial Branch Decomposition

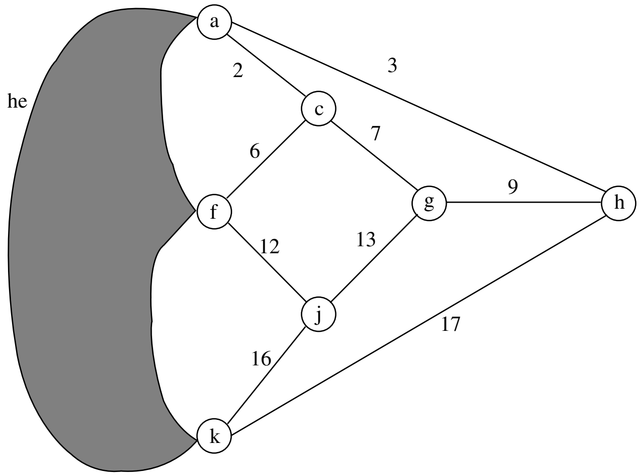

Figure 8 Example Hypergraph H^v Derived from Figure 7

branchwidth of a planar graph using the ratcatcher method is approximately $O(e^3)$. In addition, Seymour and Thomas conclude that given an integer $k \geq 0$, the complexity to find a branch decomposition of order $<k$ if one exists is $O(e^4)$. Thus, the complexity for the edge-contraction method is approximately $O(e^4)$.

6. The Cycle Method

To illustrate the cycle method, it will be applied to the partial branch decomposition illustrated in Figure 7 of the graph in Figure 1. Figure 8 is an example of H^v for nonleaf node v of the partial branch decomposition in Figure 7. The partial branch decomposition of Figure 7 is the result of a two split using the separation $(G[\{2,3,6,7,9,12,13,16,17\}], G[\{1,4,5,8,10,11,14,15\}])$ of $G\backslash\{0\}$ as the initial separation. We have omitted an illustration of the initial separation because the hypergraph associated with the initial partial branch decomposition is just the original graph.

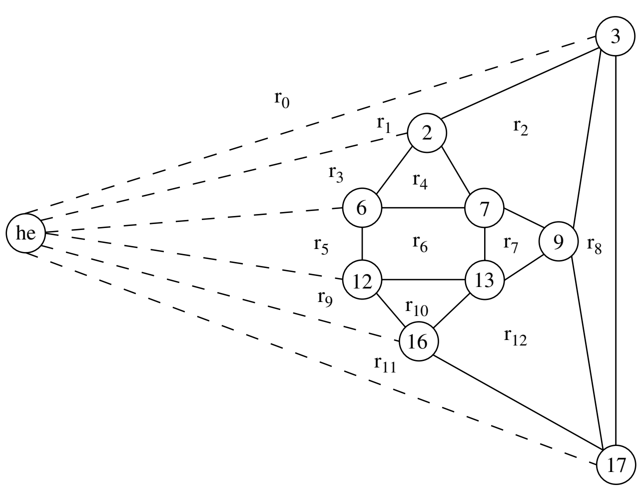

Figure 9 Medial Graph of Figure 8

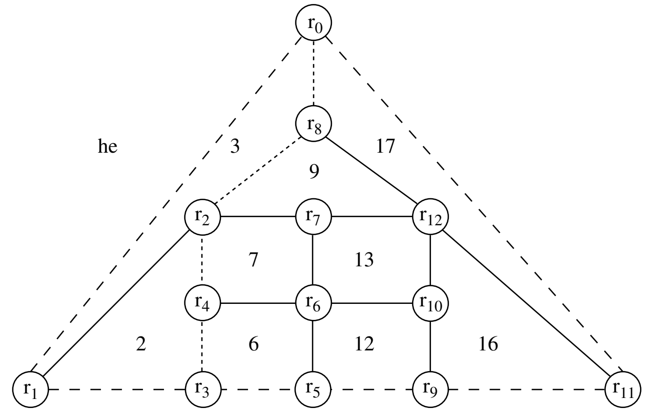

Figure 10 Corresponding Dual Graph of Figure 9

For clarification, Figure 9 illustrates the medial graph of the hypergraph in Figure 8, and Figure 10 illustrates the corresponding dual of the graph of Figure 9 with the outside region representing the hyperedge *he*. Also, notice that the regions of the graph of Figure 10 are labeled corresponding to the nodes of the graph in Figure 9 (i.e. the edges of the H^v in the hypergraph of Figure 8). Let $C \subset E(M(H^v)^*)$ denote the cycle that borders the face corresponding to the hyperedge *he*; C is illustrated in the graph of Figure 10 by the dashed cycle. The cycle method finds the shortest path between $s, t \in V(M(H^v)^*[C])$ using only the edges of $E(M(H^v)^*)\backslash E(C)$, which are the solid edges of Figure 10, such that the three cycles created by the path and C each have length at most twice the branchwidth of G. For our example, let P denote the r_0, r_3-path using the edges r_3r_4, r_2r_4, r_2r_8 and r_0r_8 in Figure 10. Notice that $P \cup C$ create three cycles, and each cycle has a length of at most eight. Also, notice that P actually partitions the regions into three sets and this corresponds to a separation $(H^v[\{2,3\}], H^v[\{6,7,9,12,13,16,17\}])$ of $H^v\backslash\{he\}$. Also, we use this separation for a two split of T if both hypergraphs resulting from the separation would have branchwidth at most the

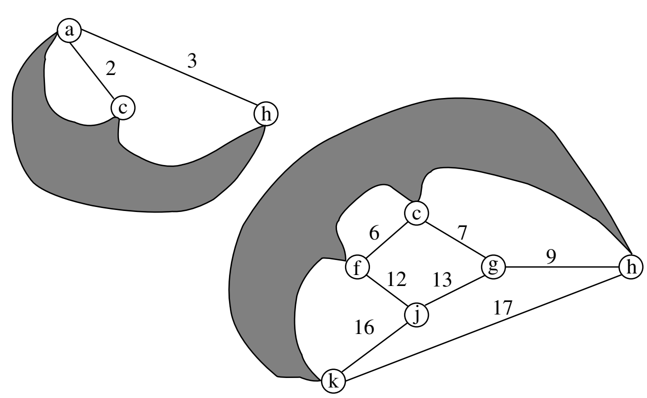

Figure 11 Resulting Hypergraphs from Using Path P

branchwidth of G. Figure 11 is an illustration of hypergraphs corresponding to the aforementioned separation. The method proceeds as follows:

While $\exists$ a node v of T with a degree greater than three
    Compute $M(H^v)$
    Let $e \in E(H^v)$
    Compute $M(H^v)^*$
    Let $C = \{f^* \mid f$ is incident to e in $M(H^v)\}$
    For each pair of distinct nodes $s, t \in V(M(H^v)^*[C])$
        Find P the shortest s, t path in $M(H^v)^* \backslash (V(C) \backslash \{s, t\})$ if one exists
        If C, C_1 and C_2, the three cycles created by $C \cup P$, have length $\leq 2\beta(G)$
        then Let (X, Y) denote the separation of $H^v \backslash \{e\}$ associated with C_1 and C_2.
            Let M_X denote the minor of $M(H^v)$ derived by contracting the edges of $M(H^v)[Y \cup \{e\}]$. Repeat the procedure for M_Y.
            If M_X and M_Y have carvingwidth at most $2\beta(G)$
            then use the separation (X, Y) for a two split to create the new partial branch decomposition T_o. Replace T by T_o and break.
        end
    end
    If no separation was found, **then** use the edge-contraction method for a separation
    end
end

Theorem 2. *Let G be a biconnected planar graph. Let (T, ν) be a extendible partial branch decomposition of G such that for all nonleaf nodes $v \in V(T)$, $M(H^v)$ is loopless and biconnected. Then (T_o, ν) the partial branch decomposition of G derived by a separation from the cycle method is extendible and for all nonleaf nodes $v \in V(T_o)$, $M(H^v)$ is loopless and biconnected.*

Proof. The ratcatcher method is used by the cycle method and the edge-contraction method to assure that the new partial branch decomposition is extendible if the previous partial branch decomposition was extendible. Thus, we need to prove the result concerning the medial graphs associated with the nonleaf nodes. Let $v \in V(T)$ be the nonleaf used to derive the separation of interest. Since we know that the edge-contraction method satisfies the result on the medial graphs, we can just focus on a separation derived by C, C_1, and C_2. Since C, C_1, and C_2 are cycles in the dual graph of $M(H^v)$, then C, C_1, and C_2 correspond to B, B_1, and $B_2 \subseteq E(M(H^v))$ bonds in $M(H^v)$. Thus, $M(H^x)$ and $M(H^y)$, the medial graphs associated with the new nodes x and y derived from the separation, will be loopless and biconnected since $M(H^v)$ was loopless and biconnected. In addition, $\mathrm{mid}(vx) = 0.5|C_1|$ and $\mathrm{mid}(vy) = 0.5|C_2|$. Notice that $M(H^x)$ and $M(H^y)$ are isomorphic to M_X and M_Y respectively. □

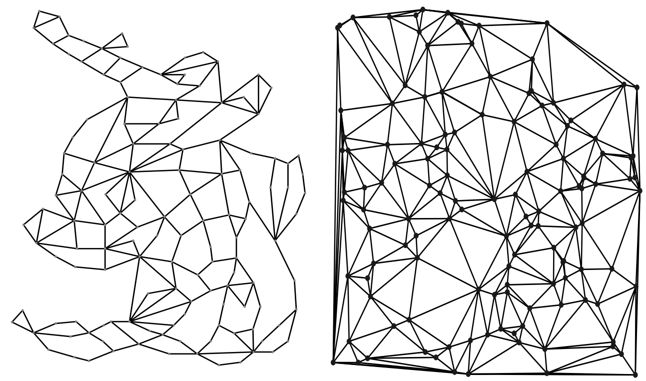

Figure 12 **Some Test Instances with Their Corresponding Branchwidth**

In summary, the cycle method creates $M(H^v)^*$ and finds a path in $E(M(H^v)^*) \backslash E(C)$ to satisfy the length constraint on the resulting cycles and the branchwidth constraint on the corresponding separation. Notice that finding cycles in the dual of the medial graphs ensures that all future medial graphs will be loopless and biconnected. Also, if the cycle method did not produce a useful separation, we would rely on the edge-contraction method for a one split of T. By the complexity of the worst case that the cycle method would find only two splits where the cardinality of X would be one, and the fact that the cycle method has to rely on the edge-contraction method, the cycle method has a running time of $O(e^4)$ where $e = |E(G)|$. In contrast, the cycle method is inherently faster than the edge-contraction method because the latter finds only one split separations where the cardinality of X

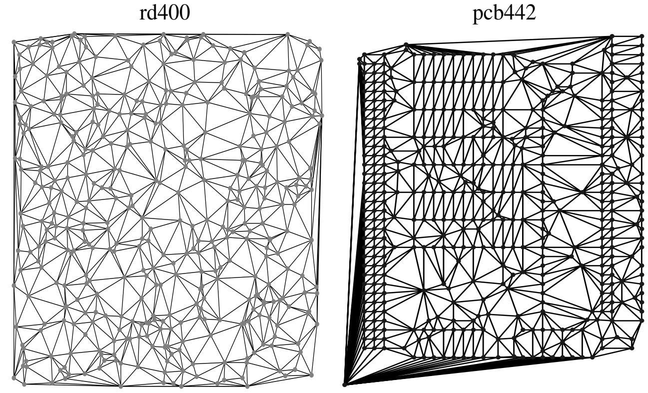

Figure 13 **Some Test Instances with Their Corresponding Branchwidth**

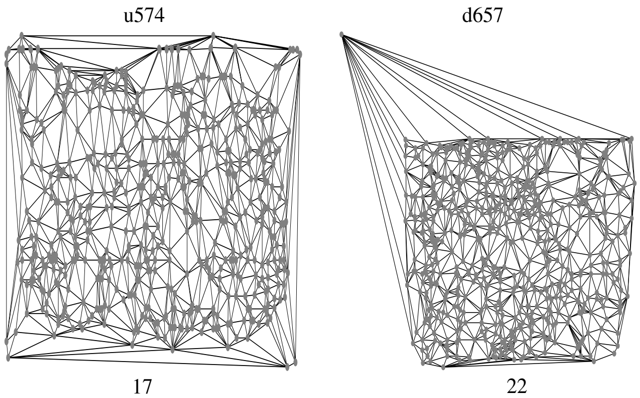

Figure 14 Some Test Instances with Their Corresponding Branchwidth

is two, while the former has the possibility of finding two splits where the cardinality of X could be greater than one.

7. Computational Results

The test instances are partitioned into three classes: telecommunications (T), Delaunay (D), and compiler (C). The members of the telecom class are maximum planar subgraphs of test instances from the telecommunications industry provided by Bill Cook at Georgia Tech. The maximum planar subgraphs were derived using a branch-and-cut scheme developed by Hicks (2004b). The test instances in the Delaunay class are Delaunay triangulations of some test instances from the TSPLIB (Reinelt 1991). For more information about Delaunay triangulations, see Edelsbrunner (1987). Figures 12, 13, and 14 illustrate some of these test instances. The test instances of the compiler class are the planar instances of control-flow graphs from actual C compilations. These test instances were provided by Keith Cooper at Rice University.

Table 1 offers a summary of the computational results by giving the ratio of the runtimes of the edge-contraction method divided by the runtimes of the cycle method for both the unaltered and greedy versions. Both the edge-contraction and the cycle method were implemented using C++. Table 2 offers a sample of the computational results to give a sense of the actual runtimes (rounded to the nearest integer) of the methods. All computations were performed on a SGI Power Challenge with 6×194 MHz processors.

Table 1 Edge-Contraction Method/Cycle Method Ratio Results

Graphs		β		Unaltered			Greedy		
class	num	min	max	min	max	geo mean	min	max	geo mean
T	7	3	5	3.5	12.0	6.48	1.0	2.0	1.41
D	40	6	22	4.3	167.27	26.82	1.31	129.42	13.02
C	115	2	4	0.9	33.73	5.51	0.97	1.02	0.99

Table 2 Cycle Method vs. Edge-Contraction Method

Graphs	Class	Nodes	Edges	$\beta(G)$	EC	Cycle	EC greedy	Cycle greedy
tele39	T	39	81	4	5	1	0.8	0.6
tele53	T	53	91	5	12	1	1	0.8
tele56	T	56	85	3	9	2	0.4	0.4
tele62	T	62	129	4	19	3	2	1
tele89	T	89	136	4	7	2	0.9	0.7
tele133	T	133	212	5	102	11	5	3
tele226	T	226	286	4	57	7	3	2
eil51	D	51	140	8	32	2	5	2
lin105	D	105	292	8	232	27	36	13
pr144	D	144	393	9	588	55	114	24
kroB150	D	150	436	10	2,434	44	412	31
pr226	D	226	586	7	1,603	235	241	51
tsp225	D	225	622	12	3,593	156	422	85
a280	D	280	788	13	1,535	72	157	51
pr299	D	299	864	11	2,043	128	404	86
p654	D	654	1,806	10	90,683	1,265	55,893	588
debflu	C	108	141	3	31	3	1	1
pastem	C	158	214	3	87	8	2	2
pdiag	C	168	221	3	116	8	2	2
rhs	C	202	249	3	364	6	3	3
deseco	C	228	301	3	411	14	4	4
iniset	C	465	465	2	102	100	2	2
parmve	C	459	591	2	2,243	128	12	12
parmov	C	583	749	3	2,967	445	28	28
parmvr	C	619	791	3	3,421	510	32	31

In addition, the runtimes of the ratcatcher method are included in the runtimes of all versions of the edge-contraction and cycle method. From the results given in the tables, the cycle method (unaltered) is an improvement to the edge-contraction method (unaltered) in all three classes. In addition, the greedy version of the cycle method is superior to the greedy version of the edge-contraction method for the telecom and Delaunay classes and equipollent for the compiler class. One explanation of this observation is that the number of edges of the graphs at each iteration of the edge-contraction method decreases by one, while the cycle method is more of a "divide-and-conquer" algorithm. Even though the worst-case analysis of the methods is the same, the cycle method finds optimal branch decompositions faster than does the edge-contraction method for most cases by factors ranging from 0.9 to 167.3. In fact, some Delaunay triangulation test instances were omitted from the computational study because the graphs were too large for the edge-contraction method (unaltered) to handle in a reasonable time frame (150,000 seconds). In addition, the cycle method never relied on the edge-contraction method for a separation in any of the test instances.

8. Conclusions and Future Work

In conclusion, the cycle method is a practical improvement over the edge-contraction method. Even though the worst-case analysis of the methods is the same, the cycle method found optimal branch decompositions faster than the edge-contraction method for most cases by factors ranging from 0.9 to 167.3. In fact, some test instances for the comparison had to be shortened because the edge-contraction method would take a considerable amount of computing time on those instances.

For future work, there may exist a way to utilize the medial graph of graphs embedded on a higher genus to find the branchwidth of those graphs. Also, the construction of a proof that an s, t-shortest path works for the cycle method if and only if any s, t-shortest path will work for the cycle method is another direction for future work. The latter task would prove that the cycle method does not have to rely on the edge-contraction method. In addition, another direction for future work is to find a polynomial-time algorithm to compute the treewidth of planar graphs. Finally, there may be a way to find an optimal branch decomposition of a planar graph with lower complexity than the edge-contraction and cycle method. Recent results by Tamaki (2003) may offer insight toward this challenging problem.

Acknowledgments
The author would like to acknowledge Bill Cook for his guidance and patience. The author would also like to thank Nate Dean and Paul Seymour for conversations about the subject. In addition, this effort could not have been accomplished without the planarity-testing codes provided by David Applegate and John Boyer (see Boyer and Myrvold 1999 for a description of the algorithm). Applegate's code was used for earlier versions of the methods, while Boyer's code was used for the recent versions of the methods. This research was partially supported by NSF Grant DMI-0217265, ONR Grant N00014-98-1-0014, and by Texas ATP Grant 99-3604-0034. The author was also supported by the AT&T Labs Fellowship Program as a graduate student in the Computational and Applied Mathematics Department at Rice University. Finally, the author would like to thank the anonymous referees that gave valuable advice for the presentation of the results.

References

Alvarez, C., R. Cases, J. Diaz, J. Petit, M. Serna. 2000. Routing tree problems on random graphs. Technical report LSI-01-10-R, Software Department, Universitat Politecnica de Catalunya, Barcelona, Spain.

Arnborg, S., J. Lagergren, D. Seese. 1991. Easy problems for tree-decomposable graphs. *J. Algorithms* **12** 308–340.

Boyer, J., M. Myrvold. 1999. Stop minding your P's and Q's: A simplified O(n) planar embedding algorithm. *Tenth Annual ACM-SIAM Sympos. Dicrete Algorithms.* ACM, New York, 140–146.

Cook, W., P. Seymour. 2003. Tour merging via branch-decomposition. *INFORMS J. Comput.* **15** 233–248.

Edelsbrunner, H. 1987. *Algorithms in Combinatorial Geometry*. Springer-Verlag, Berlin, Germany.

Geelen, J., A. Gerards, G. Whittle. 2002. Branch width and well-quasi-ordering in matroids and graphs. *J. Comb. Theory, Series B* **84** 270–290.

Hicks, I. V. 2000. *Branch Decompositions and their Applications*. Ph.D. thesis, Computational and Applied Mathematics Department, Rice University, Houston, TX.

Hicks, I. V. 2002. Branchwidth heuristics. *Congressus Numerantium* **159** 31–50.

Hicks, I. V. 2004a. Branch decompositions and minor containment. *Networks* **43** 1–9.

Hicks, I. V. 2004b. New facets for the planar subgraph polytope. Working paper, Industrial Engineering Department, Texas A&M University, College Station, TX.

Hicks, I. V. 2005. Planar branch decompositions I: The ratcatcher. *INFORMS J. Comput.* **17** 402–412.

Liebers, A. 2001. Planarizing graphs: A survey and annotated bibliography. *J. Graph Algorithms Appl.* **5** 1–74.

Reinelt, G. 1991. TSPLIB—A traveling salesman library. *ORSA J. Comput.* **3** 376–384.

Robertson, N., P. Seymour. 1985. Graph minors: A survey. *Survey in Combinatorics*. Cambridge University Press, Cambridge, UK, 153–171.

Robertson, N., P. Seymour. 1991. Graph minors X: Obstructions to tree-decompositions. *J. Combin. Theory, Series B* **52** 153–190.

Robertson, N., P. Seymour. 1995. Graph minors XIII: The disjoint paths problem. *J. Combin. Theory, Series B* **63** 65–110.

Seymour, P., R. Thomas. 1994. Call routing and the ratcatcher. *Combinatorica* **14** 217–241.

Tamaki, H. 2003. A linear time heuristic for the branch-decomposition of planar graphs. Technical report MPI-I-2003-1-010, Max-Planck-Institut Fur Informatick, Saarbrücken, Germany.