MINIMUM DOMINATING CYCLES IN OUTERPLANAR GRAPHS

by

Andrzej Proskurowski and Maciej M. Sysło

Abstract

A dominating cycle of a graph lies at distance at most one from all the vertices of the graph. The problem of finding minimum size such cycle is proved hard even when restricted to planar graphs. An efficient algorithm solving this problem is given for the class of outerplanar graphs, in which all vertices lie on the exterior face in a plane embedding of the graph.

+Department of Computer and Information Science, University of Oregon, Eugene, Oregon.
#Computer Science Department, Washington State University, Pullman, Washington (on leave from Institute of Computer Science, University of Wrocław, Wrocław, Poland).

1. Introduction

A notion of domination in graphs is closely related to the problem of distributing resources in networks. A subset D of vertices of a graph G=(V,E), $D \subseteq V$, is said to dominate the set of vertices V iff any vertex not in D is adjacent to a vertex in D. In this paper we consider a dominating cycle of a graph. A dominating cycle is a dominating set of vertices $v_0, v_1, \ldots, v_k$ of the graph which form a cycle (vertices v_i and v_{i+1} are adjacent, for $0 \leq i < k$ and addition modulo k+1). We are interested in the problem of algorithmically determining a minimum dominating cycle of a given graph G, i.e., a cycle of minimum size among all dominating cycles of G. We will show that this problem is NP-complete for planar graphs, which calls for investigation of some more restricted classes of graphs. In [3], a solution has been proposed for a class of graphs including maximal outerplanar graphs, which are triangulations of a polygon. There the reader may also find a brief discussion of the literature on the subject of domination. We will improve the technique of the algorithm presented in [3] which will allow us to find efficiently a minimum dominating cycle for any 2-connected outerplanar graph. An outerplanar graph is defined as a planar graph in which all vertices lie on the exterior face in a plane embedding of the graph. Our algorithm makes use of a tree structure that can be uniquely associated with a plane embedding of any outerplanar graph and which can be efficiently recovered from the graph's representation. Using the strategy of "pruning" this associated tree we obtain the overall time complexity proportional to the size of the original graph.

In our presentation we will employ the standard notions of a Hamiltonian cycle, which is a cycle consisting of all vertices of a graph, and an ordered rooted tree, [2], a plane embedding of a tree with a node designated as a root. To distinguish between elements of outerplanar graphs and the associated with them trees we will refer to vertices of the graphs and nodes of the trees. For a non-root node u of a rooted tree, the first node on the unique path to the root is called the father of u, of whom u is a son. All the other nodes adjacent to this father node and not on the path are called brothers of u. The counter-clockwise ordering of brothers as viewed from their common father determines seniority, with brothers encountered in their increasing seniority. Any particular embedding in the plane of an outerplane graph defines an outerplane graph.

2. Complexity of finding a minimum dominating cycle

We will now show that the problem of finding whether a dominating cycle of at most the given size exists in a general graph is NP-complete. We will demonstrate it by proving that even for graphs restricted to planar graphs the problem is likewise hard.

Problem : Dominating cycle in a planar graph, PLAN.DOM.CYCLE.
Instance: A planar graph G and an integer k.
Question: Is there a simple cycle of G of length k or less which dominates G?

We first observe that the problem is obviously in NP since the domination can be checked in time polynomial in the size of the given graph. A non-deterministic algorithm would involve, for a given graph G and an integer k, choosing a sequence of k vertices and checking that they constitute a dominating cycle of G. This decision problem has an immediate polynomial time reduction to the optimization type problem of finding a minimum size dominating cycle of G.

We will prove the NP-completeness of PLAN.DOM.CYCLE by a reduction from PLAN.HAM.CYCLE, the problem of determining the existence of a Hamiltonian cycle in a planar graph. This latter problem is known to be NP-complete, [1].

Theorem 2.1 PLAN.DOM.CYCLE is NP-complete.

Proof We present a polynomial time construction which for any instance of PLAN.HAM.CYCLE gives an instance of PLAN.DOM.CYCLE with the same answer as the original problem. Let $G=(V,E)$ be a planar graph with n vertices and m edges for which the existence of a Hamiltonian cycle is to be established. We construct a planar graph $G'=(V',E')$ with $n+3m$ vertices and $7m$ edges. Besides the vertices v' in V' corresponding directly to the vertices in V, $\{v' \mid v \text{ in } V\} \subseteq V'$, for each edge $e=(u,v)$ in E we introduce vertices u'_e, v'_e, and w'_e with the following adjacencies in E':

$\{(u',u'_e),(u',w'_e),(u'_e,w'_e),(v',v'_e),(v',w'_e),(v'_e,w'_e),(u',v')\} \subseteq E'$.

The construction is illustrated in Figure 1.

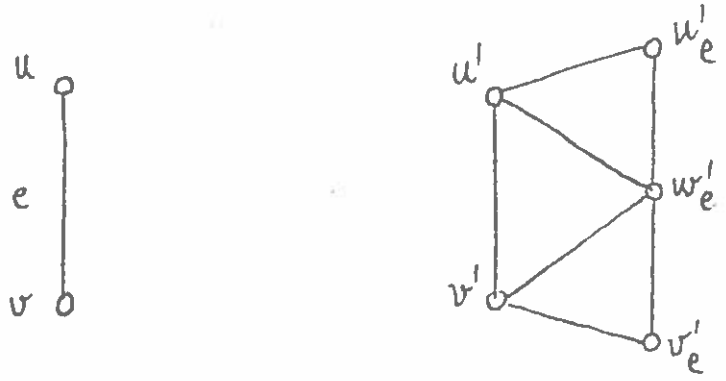

Figure 1 Construction reducing PLAN.HAM.CYCLE to PLAN.DOM.CYCLE

We prove now that the graph G' has a dominating cycle of size n iff G has a Hamiltonian cycle. To this end, we show that a dominating cycle in G' need only use the vertices v' of G' which are copies of those in G. Any dominating cycle of G' has to contain all such vertices v' in order to cover the vertices $\{v', w' | v \text{ in } V, e \text{ in } E\}$. But also, any such cycle containing vertices v' or w' can be shortened by dropping these without impairing the domination property. Thus, a dominating cycle $\langle v'_1, v'_2, \ldots, v'_n \rangle$ of G' determines a Hamiltonian cycle $\langle v_1, v_2, \ldots, v_n \rangle$ of G. The implication in the other direction is immediate. Since PLAN.HAM.CYCLE is NP-complete, a polynomial time algorithm solving PLAN.DOM.CYCLE would lead to a polynomial time algorithm solving any problem in NP. Together with the previous observation, it proves that PLAN.DOM.CYCLE is NP-complete. □

3. Plane tree associated with an outerplane graph

Since attempts to find an efficient algorithm for solving PLAN.DOM.CYCLE may be a hopeless task, we will restrict our attention to a subclass of planar graphs, outerplanar graphs. First we will

define a tree representing these graphs and give an algorithm computing this representation from a plane embedding of a given outerplanar graph.

Given a two-connected outerplane graph G, the associated plane tree T(G) is defined as follows. The internal nodes of T(G) correspond to the interior faces of G, the finite regions of the plane bound by G's edges. The external nodes of T(G) all correspond to the exterior face of G. Edges of T(G) correspond uniquely to the edges of G, and two nodes of T(G) are adjacent iff the corresponding two faces of G share an edge. Thus, edges incident to the external nodes of T(G) correspond uniquely to edges of the Hamiltonian cycle of G. Because every vertex of G lies on the exterior face, it is easy to see that T(G) has no cycles. The non-separability of G ensures connectedness of T(G), which is thus a tree. We want to preserve the information of the embedding of G and therefore require that the nodes adjacent to an interior node of T(G) be ordered in the same manner as the corresponding faces of G.

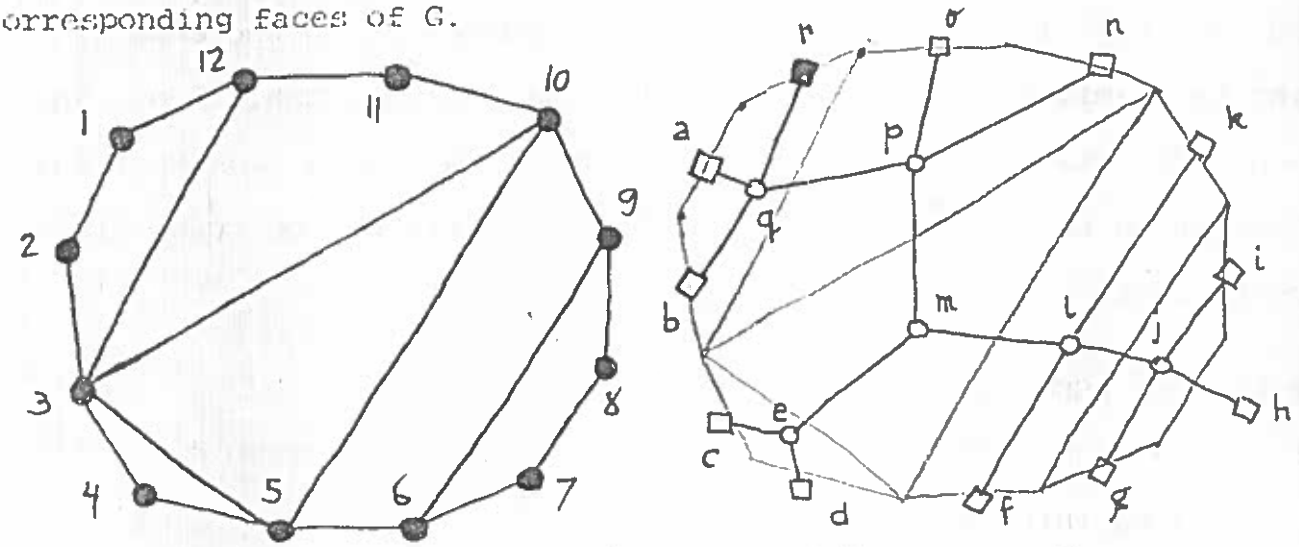

Figure 2 An outerplanar graph G and its associated tree T(G).

Our algorithm for constructing the associated tree for a given outerplane graph G assumes a plane adjacency representation of G in which, for each vertex v, the vertices adjacent to it are listed in the counter-clockwise order. Such a representation can be produced by any outerplanarity testing algorithm (for an efficient outerplanarity test see, for instance, [4]). The algorithm given below is a simplification of an algorithm which finds the dual graph of a given plane graph presented in [5]. Our algorithm traverses the Hamiltonian cycle of G creating new nodes of T(G) and "pruning" the interior faces of G, all vertices of which have been traversed, by merging them into the exterior face of G. The pruning operation is facilitated by maintaining a stack of vertices with degree greater than 2. The algorithm produces a rooted representation of T(G), which requires a stack of nodes which are the youngest among their siblings. A linear list consisting of nodes in the postorder traversal of T(G) allows for easy computation of links within "families" (father, younger and older brother, youngest and oldest son). The resulting ordered tree is rooted at the leaf (external node) corresponding to the edge of G closing the Hamiltonian cycle. We assume operations Push, Pop, and Top on the stacks, InsertNew (creating a new node), LinkSons, and LinkBrother on the list, and Next, Delete, and Degree on the plane adjacency list.

Algorithm 3.1 ASSOCIATED TREE

Input: Plane adjacency representation of an outerplane graph G
and an initial vertex s.

Output: Ordered rooted tree represetation of the associated tree T(G).

Method:

```
[0. Initialize] InsertNew(x); Push(x,Nstack); u:=Next(s);
                Push(s,Vstack); Delete(s,u); current:=u;
[1. Prune ]     while current≠s do
[1.1 a path ]       while Degree(current)<2 do
[1.1.1]                 InsertNew(x); u=Next(current);
                        Delete(current,u);
[1.1.2]                 current:=u; [end of path]
[1.2 clean-up ]     repeat [until no back edges from current]
[1.2.1]                 t:=Top(Vstack);
[1.2.2]                 if Next(t)=current then
[1.2.2.1]                       InsertNew(x); Delete(current,t);
[1.2.2.2]                       LinkSons(x); Pop(Nstack);
[1.2.2.3]                       if Degree(t)>0 then Push(x,Nstack);
                                [end of pruning a face]
[1.2.3]                 if t=s and Degree(t)=0 then
                                                or
                                        Pop(Vstack); LinkBrother(x); Push(x,
                        until t=s or Degree(t)≥1;
[1.3 current ]      if Degree(current)≥2 then
[1.3.1]                 Push(current,Vstack);
[1.3.2]                 InsertNew(x); Push(x,Nstack);
[1.3.3]                 u:=Next(current); Delete(current,u);
[1.3.4]                 current:=u;
                        [end higher degree]
                    [end of pruning]
[2. Root ]      LinkSons(x); InsertNew(x);
                [end of ASSOCIATED TREE]
```

The algorithm is illustrated in Appendix by its execution on the graph in in Figure 2. For discussion of correctness and complexity of the original algorithm the reader is referred to [5].

4. Minimal dominating subgraph of an outerplane graph

We are now ready to compute a minimal dominating cycle of an outerplanar graph represented by its associated tree. This extends the results of Proskurowski [3] which deal with maximal outerplanar graphs. We observe that representing a maximal outerplanar graph by its associated tree, as defined above, is equivalent to its representation by the weak dual tree with inclinations of nodes, used in [3]. In a rooted associated tree, the position of an internal node implies its inclination by considering position of its sibling, relative to that of their common father. Using the associated tree to represent an outerplanar graph, we can apply the technique of "pruning" this tree to obtain a representation of a minimal dominating subgraph of the graph.

Let us call a node of a tree pendant if all but one of its neighbors are leaves. In the associated tree T(G) of a given outerplane graph G, pendant nodes correspond to pendant faces of G, which have only one (base) edge in common with the rest of the graph. Pruning a pendant node u in T(G) (i.e., deleting all adjacent leaves) corresponds to deletion of all vertices of the corresponding face of G except for the base vertices. This is equivalent to merging of the pendant face into the exterior face, which results in a new outerplanar graph $G' \subseteq G$, represented by the pruned associated tree.

For the remainder of our paper we will discuss only two-connected outerplanar graphs. Let us define after [3] a fan in an outerplanar graph G to be a maximal triangulated subgraph F of G such that all triangles of F have one vertex in common and an edge incident to this vertex is the only edge F has in common with the rest of G. We call this edge base of the fan. A pivot vertex of the base dominates all vertices of F. Therefore, the nodes of the associated tree T(G) corresponding to the faces (triangles) of a fan can be pruned without a corresponding loss of this dominating vertex. A fan is represented in rooted T(G) by a maximal path of internal nodes with only left (or with only right) brothers, ending with a node with two external sons (see Figure 3(a)). The node on the other end of this path is incident to the edge representing base of the fan.

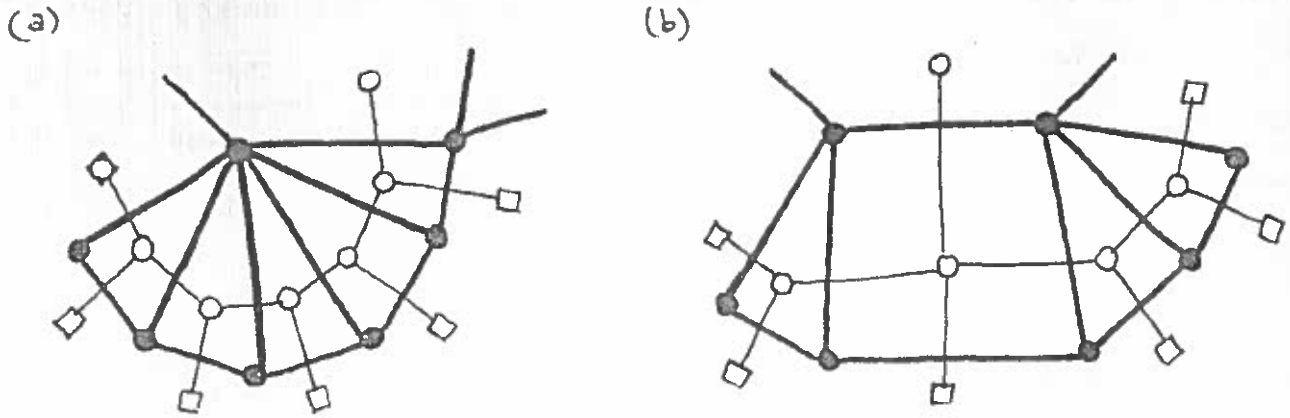

Figure 3 (a) A fan, (b) a quadrangle with fans, and their representation in the associated tree.

In addition to pruning nodes representing a fan, we will also prune nodes which represent certain quadrangle faces of an outerplanar graph. Adjacent to such a face may be at most two fans completely dominated by vertices of the only edge the face has in common with the

rest of the graph (see Figure 3(b)). Vertices of such a subgraph, which we call a shell, are dominated by the vertices of this base edge of the shell. These vertices have to be included in a dominating 2-connected subgraph of G. We state formally properties of these subgraphs of outerplanar graphs.

Lemma 4.1 Given an outerplanar graph G, not a fan or a shell, every 2-connected, dominating, outerplanar subgraph of G must contain a pivot vertex of every fan and both basic vertices of every shell of G.

Proof Let H be a fan or a shell of G. Vertices of H can be dominated either by vertices of its base, or by other vertices of H including a non-basic vertex u. In the latter case, any dominating vertex set of G has to contain also a vertex v in G-H. The base of H separates u and v, and thus any 2-connected subgraph of G containing both u and v must also contain vertices of the base of H. □

Fans and shells of a given outerplanar graph G can be easily recognized in the associated tree T(G). The pruning algorithm, to be presented shortly, processes nodes of the tree by marking the pendant nodes and pruning the leaf nodes of the current tree, i.e., the subtree of not yet pruned vertices of T(G). The mark of a node carries information about the shape of its pruned subtree: N(il) is the mark of a leaf; T(riangle) is the mark of a node corresponding to a pendant triangle face; L(eft) and R(ight) mark respectively nodes

corresponding to triangles of a fan dominated by the left and the right basic vertex; B(ase) marks a node which represents a face containing basic vertices of a fan or a shell. These definitions and Lemma 4.1 justify the following statement.

Lemma 4.2 Given an outerplanar graph G and a current subtree T' of its associated tree T(G). A pendant node of T' is marked to reflect domination of vertices of G depending on marks of its neighbors. (The conditions below indicate all marked neighbors, sons, of the pendant node.)

(i) a T-node has both sons marked N;

(ii) an L-node has two sons: left marked L or T , and right marked N;

(iii) an R-node has two sons: left marked N, and right marked R or T;

(iv) a B-node results from one of two cases. (a) It has two non-compatible sons, i.e., marked L and R, T and R, L and T, or T and T for left and right son, respectively. (b) It has three sons marked by either of the following: N,N,N; T,N,N; T,N,T; L,N,N; L,N,T; N,N,R; N,N,T; T,N,R; L,N,R, for left, middle, and right son marks, respectively.

After the pruning of T(G) has been completed, the final current tree T' represents a minimal, 2-connected, dominating, induced, outerplanar subgraph G' of G. G' is minimal in the sense that no edge or vertex of it can be removed without creating a graph which is either separable, or not an induced subgraph of G, or not dominating

G.

Algorithm 4.3 Minimal Dominating Cycle of an Outerplanar Graph.

Input: The tree T associated with a given embedding of the outerplanar graph G.

Output: The tree T' associated with an induced outerplanar subgraph C' of G dominating C.

Method: [prune leaves of the current tree marking the pendant nodes]

[0. Initialize] Set T' to T;

Mark leaves of T' by Nil;

[1. Prune] While there is a pendant node w of T'

with all sons marked

Case marks of sons of w [as in Lemma 4.2]

(i): set mark of w to Triangle;prune its sons;

(ii): set mark of w to Left; prune son of w;

(iii):set mark of w to Right; prune son of w;

(iv): set mark of w to Base; prune sons of w;

other: delete marks of sons of w;

[end of pruning]

[end of MINIMAL]

Theorem 4.4 Given a tree T associated with an outerplane graph G, the subtree T' of T output by Algorithm 4.3 is associated with a minimal, 2-connected, dominating, induced, outerplanar subgraph C' of G.

Proof By Lemmas 4.1 and 4.2, vertices of G' dominate G. Removal of any single vertex of G' which is not a tip of a pendant triangle face of G' would destroy nonseparability of G'. A pendant triangle face of G' must be represented in T' by a node whose son has been marked Base, or else the face would have been pruned during execution of Algorithm 4.3 (by Lemma 4.2). As such, its non-basic vertex dominates some deleted vertex of G not dominated by any other vertex of G'. Similarly, removal of the exterior edges of a pendant quadrangle face of G' would affect a vertex necessary to dominate some deleted vertex of G, as the node of T' representing such a face must have a son marked Base in order to be preserved. No edges of a larger pendant face of G' can be removed without impairing domination of some of its vertices. Thus, G' is minimal. □

Not every minimal 2-connected dominating outerplanar subgraph G' of G has the least number of vertices (and thus, the shortest Hamiltonian cycle). In a special case of a quadrangle face of a shell dominating the rest of the graph, there may be an adjacent to it triangle face also dominating the graph, see Figure 4(a). This is, however, the only such case.

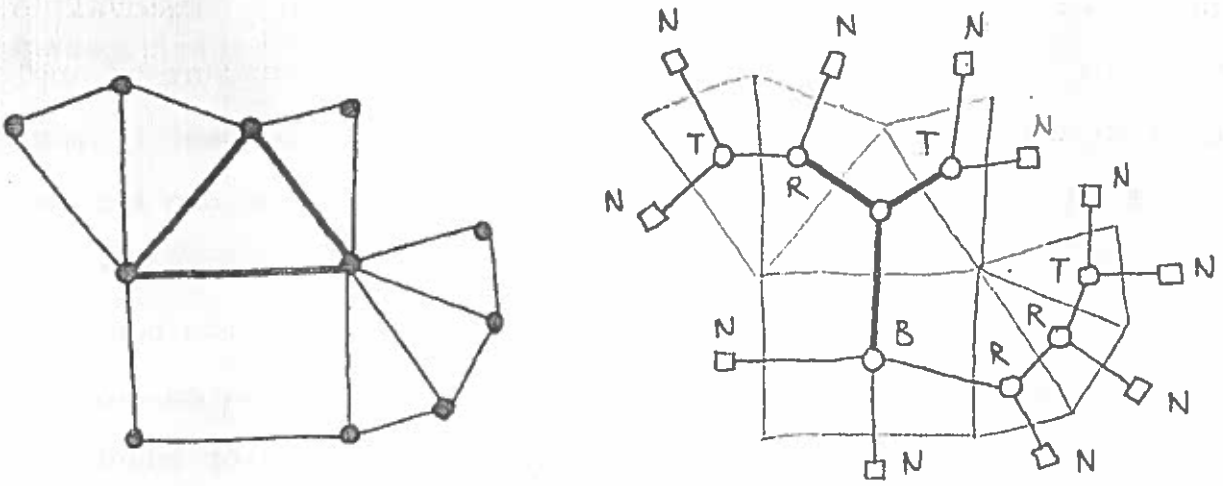

Figure 4 (a) An outerplanar graph G and (b) its associated tree T(G) with nodes marked by Algorithm 4.3.

Theorem 4.5 The Hamiltonian cycle of the dominating outerplanar subgraph G' of G produced by Algorithm 4.3 is the minimum dominating cycle of G, with the possible exception of the following case: G consists of a triangle face whose sides are bases of a shell, and two mutually non-compatible (possibly empty) fans.

Proof If G' is a triangle, it is also a minimum dominating cycle of G. If G' consists of more than one face, then all these faces are necessary to dominate G, or else they had been pruned (Theorem 4.4). If G' is a cycle of size greater than 4, then it separates any other faces of G which could jointly dominate G'. Only a G' which is a four-cycle can be dominated by an adjacent to it face of smaller size. This smaller face, C, is a triangle and must dominate the rest of G, which is also dominated by G'. This is possible only if C is adjacent to two fans, any of which can be empty or a triangle, dominated by the base of G'. □

Thus, except for an easily verifiable case, Algorithm 4.3 computes correctly a minimum dominating cycle of a given outerplanar graph. This computation is efficient.

Theorem 4.6 Algorithm 4.3 has the time complexity proportional to the size of its input graph.

Proof The associated tree T(G) of an outerplanar graph G has less nodes than G has edges. Each node of T(G) is referred to at most twice: once when assigned mark, and second time when this mark is examined, and the node is processed, both constant time operations. Once the mark is deleted or the node is pruned, the node is never considered again. Hence, the linear complexity. □

An example of marks produced by Algorithm 4.3 is given in Figure 4. A resulting marking of the associated tree and the corresponding minimum dominating cycle are indicated by bold lines. Another order of pruning the associated tree could yield a minimal dominating four-cycle.

4. Rerooting an ordered tree

An elaboration of Algorithm 4.3 computing a minimal dominating cycle of G by pruning its associated tree T(G) may use the rooted representation of T(G), as the one produced by Algorithm 3.1. In such case, it would be advantageous to have the root node of T(G) represent a face of a dominating outerplanar induced subgraph of G. In this section, we discuss the necessary rerooting algorithm (cf. [3]).

If we wish to change the location of the root of an ordered tree so that the plane embedding of the tree will not change, we have to "reroot" it. This operation involves changing the direction of father links for a number of nodes. In many respects it resembles a series of rotations (see, for instance, [2]) of the relevant nodes. Beside the change of the father link, the affected lists of brothers must be updated and the seniority order reevaluated. We give the rerooting algorithm assuming associated with each vertex: doubly linked list of sons in the seniority order, pointers to the oldest and youngest sons, pointers to older and younger brothers, and the father pointer (for the root this pointer is null). The algorithm is executed as a series of rotations of nodes on the path from the old root node toward the newly designated one. Figure 5 illustrates an individual rotation.

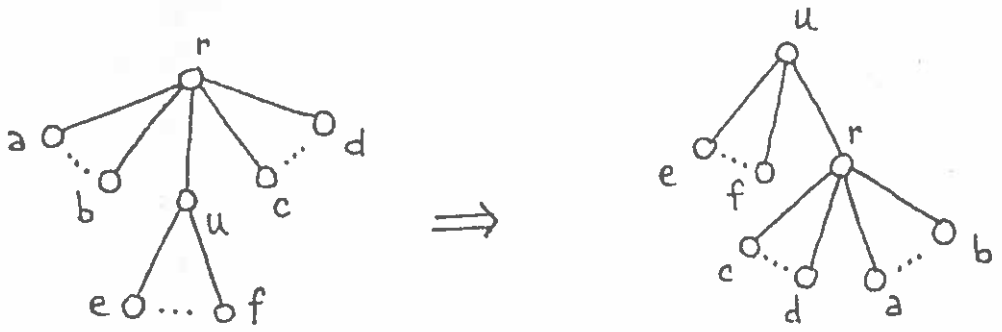

Figure 5 Changing position of the root from r to its neighbor u.

Knowledgment and Conclusion

Our investigations have been spurned by Steve Hedetniemi's list of current research problems (S.T.Hedetniemi, unpublished memo). The construction used in our NP-completeness proof results in a connected graph. It would be interesting to explore complexity of the dominating cycle problem for 3-connected graphs.

References

[1] M.R.Garey and D.S.Johnson, Computers and Intractability, Freeman, 1978.

[2] D.E.Knuth, The Art of Computer Programming, vol.I, 2nd edition, Addison-Wesley, 1973.

[3] A.Proskurowski, Minimum dominating cycles in 2-trees, Int. J. Comp. Info. Sci., 8, 5(1979), 405-417.

[4] M.M.Sysło, Outerplanar graphs: characterizations, testing, coding, and counting, Bull. Acad. Polon. Sci., Ser. Sci. Math. 26 (1978), 675-684.

[5] M.M.Sysło, An efficient cycle vector space algorithm for listing all cycles of a planar graph, Proc. Int. Coll. Graph Theory, Szeged 1978, North-Holland, 1980.

Appendix

Computation of the associated tree of the outerplane graph given in Figure 2. Assuming the initial vertex 1, the stacks Vstack and Nstack have the following contents (top of the stack is on the right-hand side).

current	Vstack	Nstack
2	1	a
3	1 3	a c
4		
5		a
		a e
	1 3 5	a c f
6	1 3 5 6	a e f g
7		
8		
9		a e f
	1 3 5	a c
		a c k
10		a e
	1 3	a
		a n
11		
12	1	a

Appendix

Computation of the associated tree of the outerplane graph given in Figure 2. Assuming the initial vertex 1, the stacks Vstack and Nstack have the following contents (top of the stack is on the right-hand side).

current	Vstack	Nstack
2	1	a
3	1 3	a c
4		
5		a
		a e
	1 3 5	a e f
6	1 3 5 6	a e f g
7		
8		
9		a e f
	1 3 5	a e
		a e k
10		a e
	1 3	a
		a n
11		
12	1	a