CIS-TR-81-02

COMPUTING THE LINE INDEX OF BALANCE OF SIGNED OUTERPLANAR GRAPHS

by

Arthur M. Farley and Andrzej Proskurowski

Department of Computer and Information Sciences

University of Oregon, Eugene, Oregon, 97403

Abstract

In a recent paper, Harary and Kabell discuss the problem of determining the line index of balance of a signed graph. They suggest a solution algorithm which would involve considering all spanning trees of a given graph in order to compute the index of this graph. The number of spanning trees may grow exponentially with size of the graph. We show that the problem is indeed difficult, being NP-complete in the general case. Subsequently, we give an efficient (linear time) algorithm solving the problem for outerplanar graphs with bounded size of interior faces.

1. Introduction

Signed graphs have been used in various areas of social science to model positive and negative relationships among entities [5, 6]. A (combinatorial) graph G=(V,E) consists of a set V of vertices and a set E of edges, such that each edge is incident to exactly two different vertices. A signed graph is a graph each edge of which has been labeled either '+' or '-'. A cycle in a graph is a sequence of vertices $v_0,\ldots,v_k$ such that $v_0=v_k$, no other two vertices are identical, and (v_{i-1}, v_i) is an edge of the graph $(1\le i\le k)$. The sign of a cycle in a signed graph is the product of the signs of all edges in the cycle. A signed graph is balanced if the sign of every cycle is positive. To determine the line balance of a plane signed graph G we need only check balance of its all interior faces. This follows from the fact that the interior faces of G form a cycle basis under operation of symmetric difference over the set of edges. Thus the sign of any cycle of G is given by the product of signs of its constituent interior faces. The line index (of balance) of a signed graph is the minimum number of edges which must be removed to obtain a balanced signed graph.

A marked graph is a graph each vertex of which has been labeled '+' or '-'. By the Correspondence Theorem [4, Theorem 1], for each balanced signed graph there exist two associated marked graphs (with reversed signs on the corresponding vertices), such that the sign of an edge in the signed graph is equal to the product of marks of its end vertices in the associated marked graph. Harary and Kabell give a linear algorithm [4, Sect. 3] for determining the marked graphs associated with a given balanced signed graph. The algorithm, henceforth called HK, employs the notion of a spanning tree of a connected graph G which is a connected, acyclic subgraph of G containing all vertices of G.

Let S be a spanning tree of a connected signed graph G to which algorithm HK has been applied. After vertices of S have all been marked by HK according to the signs of its edges (branches), the remaining edges (chords) of G can be classified as compatible or incompatible. An edge is compatible with the marks of its incident vertices if its sign equals the product of those marks; otherwise, the edge is incompatible. Harary and Kabel establish a theorem [4, Theorem 2] stating that the line index of a given connected signed graph G equals the minimum number of incompatible chords in a spanning subtree of a marked graph associated with G, over all such subtrees. The result suggests a straightforward algorithm for determining the line index of an arbitrary connected signed graph. Even in relatively simple classes of sparse graphs, each graph has an exponential number of spanning trees (as a function of the number of vertices in the graph) [7]. Therefore, the worst case execution time of the implied algorithm depends exponentially on the size of the input graph.

The problem of determining the line index of an arbitrary signed graph appears to be difficult, in general. In [2] we establish that indeed it is NP-complete. This result has been obtained independently by Chvátal (private communication). We will briefly present our argument. We first pose the problem as the decision problem:

Problem: Line index (of balance) of a signed graph,
LINE.INDEX;
Instance: A signed graph G and an integer K;
Question: Is there a set of at most K edges of G such that their removal results in a balanced subgraph of G?

An efficient solution to LINE.INDEX would imply an efficient algorithm to solve the original optimization problem of determining the line index of a signed graph G=(V,E). Such an algorithm would involve solving LINE.INDEX only for some values of $K<|E|$.

NP-Completeness is most often established by demonstrating polynomial-time reduction from a known NP-complete problem. In this case, it is the maximal-cut problem. Given a graph G=(V,E), a partition of V into two subsets determines a <u>cut</u>, being the set of edges each of which is incident to a vertex in each subset. For a given weight function defined on the edges, w: E-->Z (Where Z is the set of all integers), the <u>cut value</u> equals the sum of weights of the edges contained in the cut.

Problem: Value of a maximal cut, MAX.CUT;
Instance: A weighted graph G and an integer k;
Question: Is there a cut with value at least k?

MAX.CUT is NP-complete and remains difficult even when the weights of all edges equal 1 (SIMPLE.MAX.CUT problem [3, p. 210]). Therefore, if the domain of the weight function is generalized to contain two values, +1 and -1, the ensuing problem (UNIT.MAX.CUT) is likewise NP-complete. In [2] we present a polynomial time transformation from UNIT.MAX.CUT to LINE.INDEX, proving that LINE.INDEX is

NP-Complete.

2. Computing the line index of signed outerplanar graphs

Discovery that a problem is NP-complete prompts the search for efficient algorithms which either determine approximate solutions in the general domain, or determine exact solutions over restricted subdomains. We chose the latter course of action. LINE.INDEX proves to be efficiently solvable on maximal outerplanar graphs, mops. Mops are an interesting class of graphs being extremal graphs with certain favorable connectivity properties [1]. From a social science perspective, mops may represent networks of interacting ternary relationships (i.e., "triangles"). An outerplanar graph is a planar graph which may be embedded in the plane so that every vertex is on the exterior face. A maximal outerplanar graph (mop) is a 2-connected outerplanar graph such that every interior face is a 3-cycle (i.e. a triangle). Mops can be equivalently defined as follows:

(i) a triangle is a mop; and

(ii) any mop G having n>3 vertices can be obtained from a mop G' having n-1 vertices by connecting a new vertex to two adjacent vertices on the exterior face of G'.

We define an algorithm, requiring linear time and space, which determines the line index of balance of an arbitrary signed mop. The algorithm is straightforwardly adaptable to general outerplanar graphs having bounded size of interior faces. The decreased algorithmic difficulty of general problems when restricted to outerplanar graphs seems to be due to by the close relation of such graphs to acyclic graphs. For a two-connected outerplane graph, such as a mop M, the associated tree, T(M) [8],has internal nodes which correspond to the interior faces of M and external nodes, all corresponding to the exterior face of M. Each line of T(M) "crosses" an edge of M, see Figure 1. For a signed mop M we associate with each line of T(M) the sign of the edge of M it crosses.

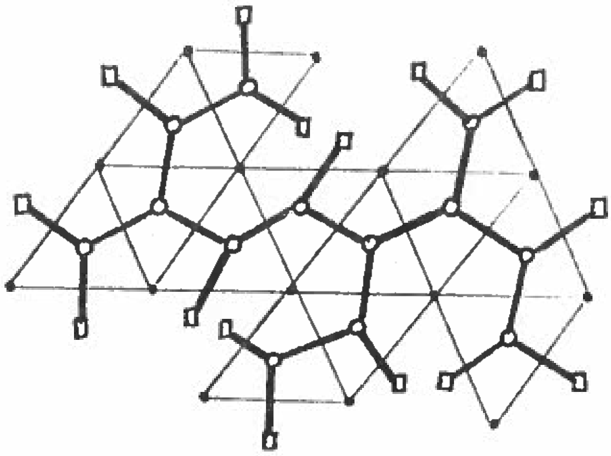

Figure 1 A mop and its associated tree

We will call a signed forest F stable if for any subset S of its internal nodes, the product of signs of edges between S and V(F)\S is positive. Let us call the product of signs of edges incident to a node the sign of that node. For any subset S of internal nodes of F, the edges of the cut defined by S constitute the symmetric difference of the sets of edges incident with each node of S. Thus, the produce of signs of the cut edges is equal to the product of sign nodes in S. Stability of a forest F is equivalent to the stability of all its node sets consisting of single nodes. It is now easy to see that a signed outerplane graph G is balanced if and only if its associated tree is stable.

In analogy to the removal of an edge in the process of balancing graphs let us define contraction of a line in a forest. This operation results in collapsing the nodes incident with the line into a new node, preserving adjacencies with other nodes. Contracting a pendant line (i.e., incident to an external node) results in a number of new external nodes. The lines incident with the other end node of the contracted line are now pendant and incident with the new nodes. The minimum number of lines that have to be contracted to stabilize a given forest F is called the stability index of F.

Theorem 1 The line index of a signed outerplane graph G is equal to the stability index of its associated tree T(G).

Proof We will prove it by induction on the line index. If G is balanced, then T(G) is stable, as the stability of an internal node of T(G) corresponds to the balance of an interior face of G. Let us assume that the theorem holds for all graphs with the line index greater to or equal to k, and let G be an outerplane graph with the line index equal to k+1. Obviously, T(G) has the stability index exceeding k, or else the inductive assumption would not hold. There is an edge e such that its removal from G results in an outerplane graph G' with line index k. The stability index of T(G') equals k and T(G') is a forest obtained from T(G) by contracting the line corresponding to e. This proves the theorem. □

Based upon this theorem, an algorithm computing the stability index of a trivalent tree can be used to compute the line index of a mop. The algorithm prunes leaf nodes of the current tree C, initially equal the given signed tree. With every pruned node the algorithm associates two values: the minimum number of lines that must be contracted to stabilize the previously pruned nodes while stabilizing the current node under assumption that the remainder of the current tree will contribute sign product of (i) plus or (ii) minus. We define an external node to be stable.

Algorithm: Stability index of a signed trivalent tree.
Input: A signed trivalent tree T.
Output: Stability Index of T.
Method:

[0.] Set C to T;
[1.] For each leaf v of T

[1.1] initialize values associated with v,

[1.2] prune v from C;

[2.] While there are more than one node in C do

[2.1] let w be a leaf of C with pruned neighbors x and y

[2.2] determine the minimum number of lines that must be contracted to keep subtrees rooted at x and y stable and

[2.2.1] w be stabilized by a "+"

[2.2.2] w be stabilized by a "-",

[2.3] prune w from C;

[3.] Let r be the last vertex in C with pruned neighbors x, y, and z;

[4.] Set StabilityIndex to the minimum number of lines which must be contracted to stabilize r, and subtrees rooted at x, y, and z.

<u>Theorem 2</u> Algorithm 1 computes correctly the stability index of a signed trivalent tree. It can be implemented to execute in time proportional to the size of its input.

<u>Proof</u> The stability of the pruned nodes of T under contraction of the minimum number of lines is invariant during the execution of the <u>while</u> loop [2]. Also, leaf nodes of the current tree C have the proper values associated with them. Considering the four cases of possible combinations of line signs for (w,x) and (w,y), values w.p and w.m can be determined in constant time based on the values for x and y.

```
Case sign (w,x) & sign (w,y) of
'-' & '-' :  w.p:=min(2+x.p+y.p, 1+x.m,y.m)
             w.m:=min(1+x.p+y.m, 1+x.m+y.p)
'+' & '-' :  w.p:=min(1+x.m+y.m, 1+x.p+y.p)
             w.m:=min(x.p+y.m, 2+x.m+y.p)
'-' & '+' :  w.p:=min(1+x.m+y.m, 1+x.p+y.p)
             w.m:=min(x.m+y.p, 2+x.p+y.m)
```

```
'+' & '+' :  w.p:=min(x.p+y.p, 2+x.m+y.m)
             w.m:=min(1+x.m+y.p, 1+x.m+y.p)
```

If a line, say (w,x), is contracted, then nodes of the subtree rooted at x remain stable according to the inductive assumption. The new node w will be stabilized by the rest of the current tree according to signs of lines incident with x and the sign of (w,y). If the end node x of the contracted line is a leaf, then w becomes a leaf and thus is stable independently from the contribution of the rest of C. The stability of the other node, y, depends on the sign of (w,y) and the values associated with the node. Upon exit from the loop [2.], the only remaining node w of C has three pruned neighbors x, y, and z. Case analysis of the signs of the three lines considering the values associated with the nodes results in a constant time determination of the StabilityIndex. The program segment performing this case analysis is given in Figure 2. □

```
Case sign (w,x)&sign(w,y)&sign(w,z)of
'-'&'-'&'-':  StabilityIndex:=min(1+x.p+y.m+z.m,
                 1+x.m+y.p+z.m, 1+x.m+y.m+z.p, 3+x.p+y.p+z.p)
'+'&'-'&'-':  StabilityIndex:=min(x.p+y.m+z.m,
                2+x.p+y.p+z.p, 2+x.m+y.p+z.m, 2+x.m+y.m+z.p)
'-'&'+'&'-':  StabilityIndex:=min(x.m+y.p+z.m,
                2+x.p+y.p+z.p, 2+x.p+y.m+z.m, 2+x.m+y.m+z.p)
'-'&'-'&'+':  StabilityIndex:=min(x.m+y.m+z.p,
                2+x.p+y.p+z.p, 2+x.p+y.m+z.m, 2+x.m+y.p+z.m)
'+'&'+'&'-':  StabilityIndex:=min(1+x.p+y.m+z.m,
                1+x.m+y.p+z.m, 1+x.p+y.p+z.p, 3+x.m+y.m+z.p)
'+'&'-'&'+':  StabilityIndex:=min(1+x.p+y.m+z.m,
                1+x.m+y.m+z.p, 1+x.p+y.p+z.p, 3+x.m+y.p+z.m)
'-'&'+'&'+':  StabilityIndex:=min(1+x.m+y.m+z.p,
                1+x.m+y.p+z.m, 1+x.p+y.p+z.p, 3+x.p+y.m+z.m)
'+'&'+'&'+'"  StabilityIndex:=min(x.p+y.p+z.p,
                2+x.p+y.m+z.m, 2+x.m+y.p+z.m, 2+x.m+y.m+z.p)
```

<u>Figure</u> <u>2</u> Case analysis for processing of the root node.

We have implemented Algorithm 2 in the programming language Pascal. Figure 3 illustrates execution of the program on the associated tree of the mop in Figure 1.

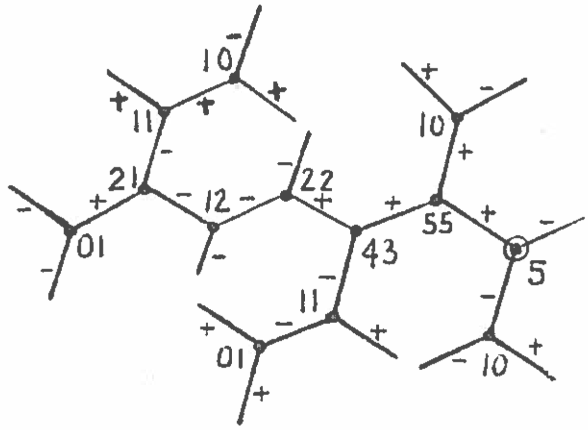

Figure 3 Values associated with nodes of a tree during execution of algorithm determining the stability index (Algorithm 1).

3. Conclusion

Algorithm 1 can be extended to apply efficiently to graphs having bounded size of interior faces by generalization of the case analyses. We are currently investigating problems related to the line index of balance. The sign index problem (minimum number of sign changes necessary to balance a signed graph) is equivalent to the line index problem, and thus NP-complete. Algorithm 1 can be modified to solve it efficiently for maximal outerplanar graphs. The vertex index problem (minimum number of vertices which have to be removed to balance a signed graph) seems likewise to be difficult.

4. References

[1] A.M.Farley, Networks immune to isolated line failures, Networks 11 (1981), to appear.

[2] A.M. Farley and A.Proskurowski, The complexity of determining the line index of a signed graph, University of Oregon, UO-CIS-TR-80/19 (1980).

[3] M.R. Garey and D.S.Johnson, Computers and Intractability, W.H. Freeman 1978.

[4] F. Harary and J.A.Kabell, A simple algorithm to detect balance in signed graphs, Mathematical Social Sci.1 (1980), 131-136.

[5] F. Heider, Attitudes and cognitive organization, J. Psych. 21 (1946), 107-112.

[6] C. Martindale, Structural balance and the rules of narrative in "les Liaisons Dangereuses", Poetics (1976), 57-73.

[7] B.R.Meyers, On spanning trees, weighted compositions, Fibonacci numbers and resistor networks, SIAM Review 12 (1974), 257-262.

[8] A.Proskurowski and M.M.Sysło, Minimum dominating cycles in outerplanar graphs, Int. J. Comp. Info. Sci., to appear.