CIS-TR-80-23

THE COMPLEXITY OF DETERMINING THE LINE INDEX OF A SIGNED GRAPH

by

Arthur M. Farley*

Andrzej Proskurowski*

Abstract

In a recent paper, Harary and Kabell [5] discuss the problem of determining the line index of balance of a signed graph. They establish a theorem which suggests a computational procedure solving this problem. The procedure is based upon a simple modification of their linear algorithm for detecting balance in signed graphs. Unfortunately, this procedure would involve considering all spanning trees of a given signed graph, the number of which may grow exponentially with the size of the graph. The question arises whether the problem of determining the line index of a signed graph can be solved efficiently (i.e., in polynomial time). We show that the problem is indeed difficult, being NP-complete in the general case.

* Department of Computer and Information Science, University of Oregon, Eugene, OR 97403.

1. Computing the index of a signed graph.

Signed graphs have been used in various areas of social science to model positive and negative relationships among entities [6,7]. A (combinatorial) graph $G=(V,E)$ consists of a set V of vertices and a set E of edges, s.t. 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 two other vertices are identical, and (v_{i-1}, v_i) is an edge of the graph $(1 \leq i \leq k)$. The sign of a cycle in a signed graph is the product of the signs of all edges of the cycle. A signed graph is balanced if the sign of every cycle is positive. 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 [5, Theorem 1], for each balanced signed graph B there exist two associated marked graphs (with reversed signs on corresponding vertices), such that the sign of an edge in the signed graph is equal to the product of signs of its end vertices in the associated marked graph. Harary and Kabell give a linear algorithm [5, Sect. 3] for determining the marked graphs corresponding to a given balanced signed graph. The algorithm, henceforth called HK, employs the notion of a spanning tree of a connected graph G. A connected graph is a graph such that a sequence of incident edges exists between any pair of vertices. A spanning tree of a connected graph G 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 Kabell establish a theorem [5, Theorem 2] stating that the line index of a given connected signed graph G equals the minimum number of incompatible chords associated with a marked spanning subtree of G, over all such marked subtrees.

The result suggests a straightforward algorithm for determining the line index of an arbitrary connected signed graph. Even in relatively simple, sparse classes of graphs, each graph has an exponential number of spanning trees (as a function of the number of vertices in the graph) [3]. Therefore, the worst case execution time of the implied algorithm depends exponentially on the size of the input graph.

2. Complexity of some decision problems.

Recent research in computational complexity has provided a framework for evaluating the inherent difficulty of given decision problems [3]. A relevant complexity concept is that of NP-completeness. It is based on the notion that a computational procedure is efficient if and only if its execution time grows no faster than a polynomial function of the size of its input. The class of problems for which a potential solution can be evaluated in polynomial time is called NP. The computational complexity of a problem in this class depends upon the number of its potential solutions and our ability to select correct solutions. There exists a large subclass of NP consisting of problems which are not known to be solvable by any efficient algorithm; however, if such an algorithm is found, it would solve efficiently all problems in NP. The problems in this class are all interrelated by polynomial time transformations and are called NP-complete problems. The question as to whether NP-complete problems are efficiently solvable or not has received much attention in the past decade, but has yet to be resolved.

In this paper, we establish that the problem of determining the line index of an arbitrary signed graph is NP-complete. We first pose the problem as the following 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 $k < |E|$. We will demonstrate NP-completeness of LINE.INDEX by presenting a polynomial time reduction from a known NP-complete problem. The problem involves determining the maximum value of a cut in a weighted graph. Given a graph G=(V,E), a partition P dividing V into two subsets determines a cut, being the set of edges each of which is incident to a vertex in each of the subsets. For a given weight function defined on the edges, $w: E \to Z$ (where Z is the set of integers), the value of the cut c(P) equals the sum of weights of 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?

It is well known that 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 contains two values, -1 and +1, the ensuing problem is likewise NP-complete. We define this special case of MAX.CUT as follows:

Problem: Value of a maximal cut, UNIT.MAX.CUT;

Instance: A graph G with edges weighted -1 and +1, and an integer k;

Question: Is there a cut of G with value at least k?

3. Main result.

We will use a polynomial time transformation from UNIT.MAX.CUT to LINE.INDEX to prove that the latter problem is NP-complete.

Theorem 1. LINE.INDEX is NP-complete.

Proof: Assume that we are given an instance of LINE.INDEX: a signed graph $G=(V,E)$ and an integer k. Let a subset F of at most k edges of G be a proposed solution to the problem. An application of algorithm HK to $G'=(V,E-F)$ can determine correctness of this claim in linear time. Thus, LINE.INDEX is in NP.

Now, suppose we are presented with an instance of UNIT.MAX.CUT: a weighted graph $G=(V,E)$ and an integer $k=k_1$. Let m denote the number of edges weighted +1 in G (wlog., $m > k_1$) and let $k_2=m-k_1$. Let us define the following construction of an instance of LINE.INDEX: for every edge e of G, label e '+' if $w(e)=-1$ and '-' if $w(e)=+1$, and establish the constant $k=k_2$. We will show that the answers to these two instances of UNIT.MAX.CUT and LINE.INDEX are identical. Let us assume that the answer to LINE.INDEX is 'yes', i.e., the removal of some set F of edges, $|F| \leq k_2$, results in a balanced graph. Then, there exists a partition P of V into subsets of vertices marked '+' and '-', such that all edges in E-F are compatible with marks of V. The value of the corresponding cut c(P) is equal to the difference between the number of edges in F weighted +1 (i.e., compatible edges signed '-') and the number of edges in F weighted -1 (i.e., incompatible edges signed '+'). Let us denote by p the number of incompatible edges signed '+' and by q the number of incompatible edges signed '-', $p+q=|F|$. Then the cut value equals

$$(m-q)-p = m-|F| \geq m-k_2 = k_1.$$

Thus, the answer to the original problem is also 'yes'. Similarly, a cut with value at least k_1 determines the marks of vertices of G so that if there are r incompatible edges signed '+', then at least k_1+r compatible edges are signed '-'. This gives the upper bound on the total number of incompatible edges (in this particular marking) $r+(m-(k_1+r)) = m-k_1 = k_2$. Hence, the answer to the corresponding instance of LINE.INDEX is also 'yes'. Thus, an efficient algorithm solving LINE.INDEX could be used to solve efficiently UNIT.MAX.CUT after transforming a given instance of the latter. As we established that LINE.INDEX is in NP, this completes our proof that LINE.INDEX is NP-complete. []

4. Conclusion.

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. LINE.INDEX proves to be efficiently solvable on maximal outerplanar graphs, mops, (to be reported elsewhere, [2].) Mops are an interesting class of graphs being minimum (in number of edges) graphs with certain favorable connectivity properties [1]. From a social sciences point of view, mops may represent networks of interacting ternary relationships (i.e., "triangles"). Mops are a subclass of planar graphs, for which the SIMPLE.MAX.CUT problem is known to be efficiently solvable [4]. This might suggest that UNIT.MAX.CUT and, by our transformation, LINE.INDEX are likewise efficiently solvable when restricted to planar graphs. Unfortunately, correctness of the known efficient solution algorithm for SIMPLE.MAX.CUT depends upon the fact that all edge weights are positive. This is not the case for UNIT.MAX.CUT. Thus, it remains an open question as to whether LINE.INDEX is efficiently solvable or NP-complete on planar graphs.

5. References.

[1] A.M. Farley, Networks immune to isolated line failures, Networks, to appear.

[2] A.M. Farley and A. Proskurowski, Computing the line index of a signed outerplanar graph, in preparation.

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

[4] F. Hadlock, Finding a maximum cut of a planar graph in polynomial time, SIAM J. Comptg, 4(1974) 221-225.

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

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

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

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