Theoretical Computer Science 131 (1994) 361–374
Elsevier

Drawing graphs in two layers

Peter Eades

Department of Computer Science, University of Newcastle, University Dr., Callaghan, NSW 2308, Australia

Sue Whitesides

School of Computer Science, McGill University, 3480 University St. 318, Montréal, Québec, Canada H3A 2A7

Communicated by F.P. Preparata
Received February 1991
Revised June 1993

Abstract

Eades, P. and S. Whitesides, Drawing graphs in two layers, Theoretical Computer Science 131 (1994) 361–374.

Let $G=(U,L,E)$ be a bipartite graph with vertex set $U\cup L$ and edge set $E\subseteq U\times L$. A typical convention for drawing G is to put the vertices of U on a line and the vertices of L on a separate, parallel line and then to represent edges by placing open straight line segments between the vertices that determine them. In this convention, a drawing is *biplanar* if edges do not cross, and a subgraph of G is *biplanar* if it has a biplanar drawing. The main results of this paper are the following: (1) it is NP-complete to determine whether G has a biplanar subgraph with at least K edges; (2) it is also NP-complete to determine whether G has such a subgraph when the positions for the vertices in either U or L are specified; (3) when the positions of the vertices in both U and L are specified, the problem can be solved in polynomial time by transformation to the longest ascending subsequence problem.

1. Introduction

For the purposes of this paper, a *bipartite graph* $G=(U,L,E)$ consists of two sets U and L of vertices and a set $E\subseteq U\times L$ of edges. Note that a particular vertex bipartition U,L is specified in the definition of G. Hence references to G presume this specified partition.

Correspondence to: S. Whitesides, School of Computer Science, McGill University, 3480 University St. 318, Montréal, Québec, Canada H3A 2A7. Email addresses: cades@cs.newcastle.edu.au and suc@cs.mcgill.ca.

*Supported in part by grants from the Australian Research Council and the Natural Science and Engineering Research Council of Canada.

SSDI 0304-3975(93)E0147-V

It is often useful to draw G so that the vertices of U (the "upper part") and L (the "lower part") are on horizontal lines λ_U and λ_L, respectively, with λ_U above λ_L; the edges are drawn as open straight line segments between their endpoints as in Fig. 1. Once the y coordinates of the horizontal lines λ_U and λ_L are specified, a *drawing* of G is defined by giving an x coordinate for each vertex, as the y coordinates of the vertices of U and L are equal to the y coordinates of λ_U and λ_L, respectively.

A drawing is *biplanar* if it has no edge crossings; a graph is *biplanar* if it has a biplanar drawing. In this paper we investigate the problem of finding, given a bipartite graph, a biplanar subgraph with a maximum number of edges. (The biplanar subgraph need not be induced.) We also study this problem when the positions of some or all of the vertices are prescribed.

Layouts of this kind have been investigated in two application areas: aesthetic layout of directed graphs [5, 7, 8, 10, 13, 17, 18, 20, 21, 23–25] and global routing for row-based VLSI layout (see, for example, [22, 26]).

Most methods for drawing directed graphs involve arranging nodes on "levels", i.e. on parallel horizontal lines, and drawing edges in between, as in Fig. 2. These methods involve a crucial "crossing reduction step" to reduce the number of edge crossings in between two successive layers. Specifically, this step involves choosing an x coordinate $x(u)$ for each vertex u belonging to the lower part L of a bipartite graph $G=(U, L, E)$ so that the number of edge crossings is minimized. This problem is NP-complete [12]

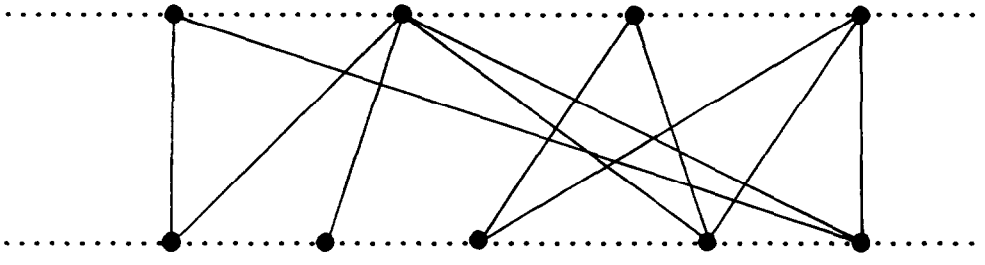

Fig. 1. A drawing of a bipartite graph.

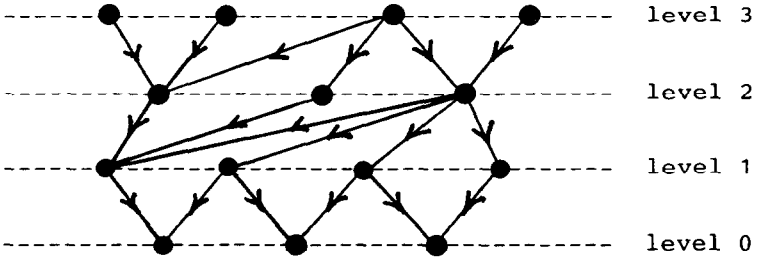

Fig. 2. Drawing a directed graph with "levels".

even when one part is held fixed [9]. The current paper examines the complexity of the corresponding problem of finding a maximum subgraph of a bipartite graph with no crossings.

In "standard cell" technology for VLSI, modules are arranged in rows with wiring channels between each pair of rows. A planar subgraph of the net list represents a subset of the set of nets that can be routed in one layer.

In Section 2 we show the following: given G and positive integer K, it is NP-complete to determine whether G has a biplanar subgraph with at least K edges. Then we point out that this problem remains NP-complete under various restrictions of the input.

In Section 3 we show the following: given G, positive integer K and positions for all the vertices in one of the sets U and L, it is NP-complete to determine whether G has a subgraph that

(a) has a biplanar drawing respecting the vertex position requirements for G and
(b) has at least K edges.

The subgraph need not be induced.

In Section 4 we show that if the positions of all vertices of G are fixed, then a biplanar subgraph with a maximum number of edges can be found in polynomial time.

2. MAXIMUM BIPLANAR SUBGRAPH is NP-Complete

Consider the MAXIMUM BIPLANAR SUBGRAPH (MBS) problem:

Instance: A bipartite graph $G=(U, L, E)$ and a positive integer K.
Question: Does G have a biplanar subgraph (not necessarily induced) with at least K edges?

Theorem 2.1. MBS *is* NP-*complete.*

To prove Theorem 2.1, a simple graph-theoretic characterization of biplanarity from [6] is needed. A *caterpillar* is a connected graph that has a path b called the *backbone* such that all vertices of degree larger than one lie on b. The edges of a caterpillar that are not on the backbone are the *legs* of the caterpillar. Without loss of generality, we shall assume that the backbone of a caterpillar is chosen so that its endpoints have degree one, i.e. they have no legs attached. A caterpillar is illustrated in Fig. 3.

Lemma 2.2 (Eades et al. [6]). *A bipartite graph is biplanar if and only if it is a collection of disjoint caterpillars.*

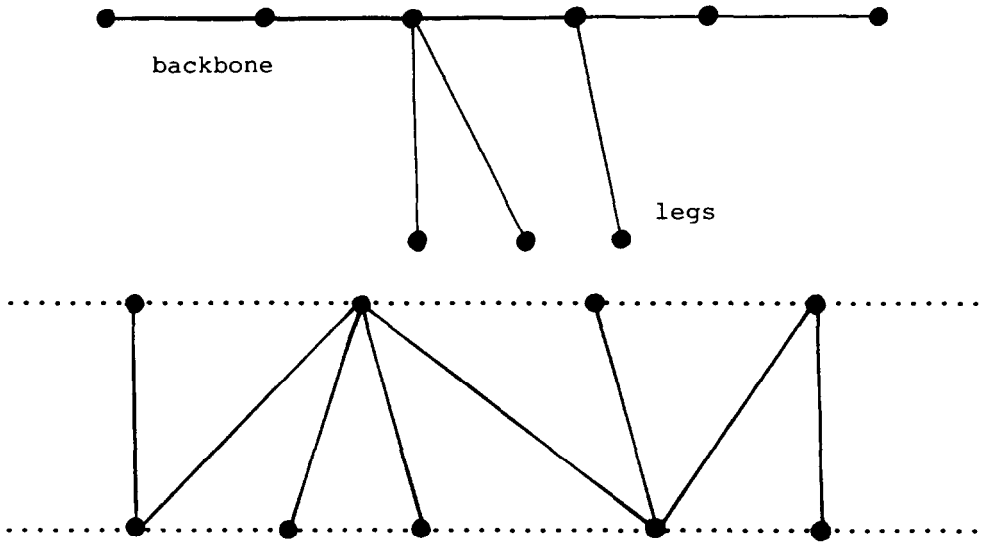

Fig. 3. A caterpillar and its biplanar drawing.

Caterpillars (and thus biplanarity) can be recognized in linear time [4, 15]. Intuitively, a caterpillar is similar to a path; this intuition leads to a proof of Theorem 2.1.

Proof of Theorem 2.1. We show that the Hamiltonian Path problem [11, p. 199] transforms to MBS. (The Hamiltonian path problem is to determine whether the n vertices of a given graph can be ordered $v_1, v_2, \ldots, v_n$ so that, for all i from 1 to $n-1$, vertices v_i and v_{i+1} are joined by an edge.)

Consider an instance $H=(V,F)$ of Hamiltonian Path, where H is a graph with n vertices and m edges. Split each edge $(a,b)\in F$ by adding a new vertex $\eta(a,b)$ between its endpoints, as in Fig. 4.

Let $G=(U,L,E)$ be the bipartite graph so formed, where

$$U=V,$$

$$L=\{\eta(a,b)\colon (a,b)\in E\}$$

and

$$E=\{(a,\eta(a,b))\colon (a,b)\in E\}\cup\{(\eta(a,b),b)\colon (a,b)\in E\}.$$

This graph can be constructed in time polynomial in the length of the description of H. Note that G has $n+m$ vertices, and let K denote $n+m-1$, which is one less than the number of vertices of G. Figures 4(a) and (b) show a graph H before and after its transformation to graph G.

Suppose that H has a Hamiltonian path $P=(u_1, u_2, \ldots, u_n)$. Each edge of H is either an edge of P or a "chord" of P that joins two nonconsecutive vertices of P. Call the

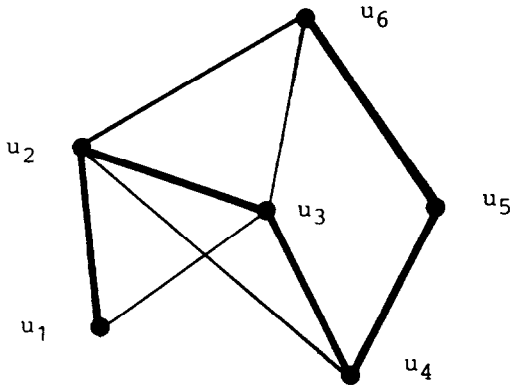

Fig. 4(a). A graph H (the ordering $u_1, u_2, \ldots, u_6$ gives a Hamiltonian path H).

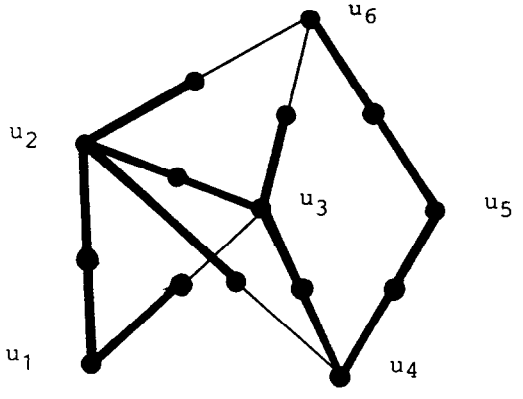

Fig. 4(b). The graph after transformation to G (the thick lines form a spanning caterpillar with $u_1, u_2, \ldots, u_6$ on the backbone).

vertices of G that belong to set L the "midpoints" of edges in H, as they are created by subdividing edges of H.

Path P can be used to obtain a spanning tree T for G as follows. The vertices of T consist of all the vertices U and L of G. The edges of T are of two types:

(1) The edges of G that come from subdividing the edges of P; in other words, for each edge (u_i, u_{i+1}) on path P in H, there are two edges $(u_i, \eta(u_i, u_{i+1}))$ and $(\eta(u_i, u_{i+1}), u_{i+1})$ of G in T.

(2) The edges of G that join the midpoint of a chord (u_j, u_k) of P to the lower-indexed endpoint of that chord; in other words, a chord (u_j, u_k) of P with $j<k$ gives rise to an edge $(\eta(u_j, u_k), u_j)$ in T.

The edges of type (1) form a simple path P' containing all the vertices of G in the set U together with all the vertices in the set L of the form $\eta(u_i, u_{i+1})$. The remaining

vertices of G are all midpoints of chords of P and so have the form $\eta(u_j, u_k)$, where $j<k$ and k is not equal to $j+1$. Each such vertex is joined to a vertex of P' by an edge of type (2) in T. Hence T is a spanning tree for G and so has K edges. In fact, T is a spanning caterpillar with backbone P'. Therefore, G contains a biplanar subgraph with K edges.

Conversely, suppose that G has a biplanar subgraph C with K edges. Since C must be a disjoint collection of caterpillars and since G has only $K+1$ vertices, C must be a single caterpillar that spans G. Let b be a maximal length backbone for C, so its endpoints have degree one. Suppose that $u \in U$; we claim that u must belong to the backbone b. If this were not so, then u would have to be adjacent to a vertex in b. Since the only neighbors of u are midpoint vertices, u would be adjacent to a vertex η of L that belongs to b. All vertices of L have degree two, so η could not be an intermediate vertex of b, as η would then have degree at least three, nor could it be an endpoint of b, as these have degree one. Therefore, the backbone b contains all the vertices of the set U. A Hamiltonian path through the corresponding vertices $V=U$ in H is obtained by listing the vertices of U in H in the same order as they appear along b in G.

The thick edges in Fig. 4(a) indicate a Hamiltonian path P for H; the thick edges in Fig. 4(b) point out a spanning caterpillar for G whose maximal backbone b contains the U vertices in the same order in which they appear in P. □

The Hamiltonian Path problem remains NP-complete under various restrictions, and these can be carried through to MBS. We give one example.

MBSR

Instance: A positive integer K and a biconnected planar bipartite graph $G=(U, L, E)$, where each vertex in U has degree 3 and each vertex in L has degree 2.

Question: Does G have a biplanar subgraph with at least K edges?

Because the Hamiltonian Path problem remains NP-complete even for regular graphs of degree 3, we have the following theorem.

Theorem 2.3. MBSR *is* NP-*complete.*

The correspondence between the Hamiltonian Path problem and MBS suggests a method for finding approximate solutions to the problem of finding a biplanar subgraph of maximum size. One can look for an approximate solution to the problem of finding a longest path (see, for example, [1, 14]), then use a long path as a large "backbone" b, adding a "leg" for each vertex that is not on b but is adjacent to a vertex on b.

3. ONE-PART-FIXED is NP-complete

In this section we show that the following problem is NP-complete.

ONE-PART-FIXED (OPF)

Instance: A bipartite graph $G=(U, L, E)$, a position $x(u)$ on λ_U for each $u \in U$, and a positive integer B.

Question: Does G contain a biplanar subgraph G' with at least B edges such that G' has a biplanar drawing with each vertex $u \in U$ at $x(u)$? (G' need not be induced.)

Theorem 3.1. OPF *is* NP*-complete and remains* NP*-complete even when the vertices of* U *have degree* 1 *and the vertices of* L *have degree at most* 2.

The proof is by reduction from the NP-complete problem VERTEX COVER [11], stated below. A subset V' of the vertex set V of a graph $H=(V, F)$ is a *vertex cover* for H if every edge $f \in F$ has at least one endpoint in V'.

VERTEX COVER (VC)

Instance: A graph $H=(V, F)$ and a positive integer $K \leqslant |V|$.

Question: Does V contain a vertex cover V' of size K or less?

Before giving a formal proof, we begin with an overview.

Overview of the proof. We construct an instance G of OPF from an instance H of VC as follows. The upper vertices of G are assigned fixed positions. A vertex v_i of degree $d(i)$ in H is represented by a "cell" in G containing $d(i)$ consecutive upper vertices. The upper vertices of one cell are separated from the upper vertices of the next cell by the upper vertices of a "wall". The upper vertices in a wall that appear between the first and last upper vertices of that wall have degree 1.

Each edge (v_i, v_j) of H is represented in G by a pair of "cover" edges incident to a common lower vertex of G. The upper vertices of these two edges belong to the respective cells for v_i and v_j and are incident to no other edges.

Each cell has a pair of "blocker" edges joining the last upper vertex of the left cell wall, and the first upper vertex of the right cell wall, to a common lower vertex. Cells contributing fewer than two blocker edges to a maximal biplanar subgraph G' of G in fact contribute one blocker edge and all their cover edges; such cells correspond to the vertices in a vertex cover for H of size at most K. Cells contributing both blocker edges correspond to vertices of H outside the vertex cover.

The smaller the size K of the vertex cover required for H, the larger the size B of the biplanar subgraph required for G. The number of wall vertices is chosen so that any maximal biplanar subgraph G' of G contains all the edges incident with an upper vertex of a wall, provided that the vertex is not the first or last upper vertex of that wall (such edges are called "wall" edges). Hence G' contains at most one of the two cover edges of G that together represent a single edge of H; G is designed so that any

maximal biplanar subgraph G' of the required size must contain both edges of many of the cell "blockers". When both edges of a blocker are used, then that cell can contribute no cover edges. Obtaining a large number of both blocker edges and cover edges requires clustering the cover edges chosen into a few cells and taking both the blocker edges for the remaining cells.

This concludes the overview, and we now proceed to the formal proof.

Proof. Suppose $H=(V, F)$ together with K is an instance of VC. From this we construct an instance of OPF, i.e.: a bipartite graph $G=(U, L, E)$, a position $x(u)$ on λ_U for each $u \in U$, and a positive integer B. We assume that H is given by listing for each vertex the edges incident to that vertex.

Denote $|V|$ by n and $|F|$ by m. Suppose that $V=\{v_1, v_2, \ldots, v_n\}$.

We create a sequence of n "cells", each corresponding to a vertex v_i, $1 \leqslant i \leqslant n$. The cells are separated by a sequence of $n+1$ "walls", as illustrated in Fig. 5.

The ith wall consists of $p+2$ upper vertices placed at consecutive positive integer values of x, p vertices in the lower part, and p edges matching these lower vertices to all but the first and last upper wall vertices (see Fig. 5). We choose $p=2(m+n)+1$. The first upper vertex of the first wall is to be placed at $x=1$. Between the last upper vertex of the ith wall and the first upper vertex of the $(i+1)$th wall, we leave a gap of $d(i)$ consecutive integer x-values, where $d(i)$ is the degree of vertex $v_i \in V$. This specifies the positions of all upper wall vertices; specifically, if u is the jth upper vertex of the ith wall (and if the empty sum arising when $i=1$ is taken to be 0), then

$$x(u)=(i-1)(p+2)+\sum_{k=1}^{i-1} d(k)+j.$$

Recall that wall edges are edges incident to an intermediate upper vertex of some wall. Since intermediate upper wall vertices each have degree 1, the total number of wall edges (taken over all walls) is $p(n+1)$.

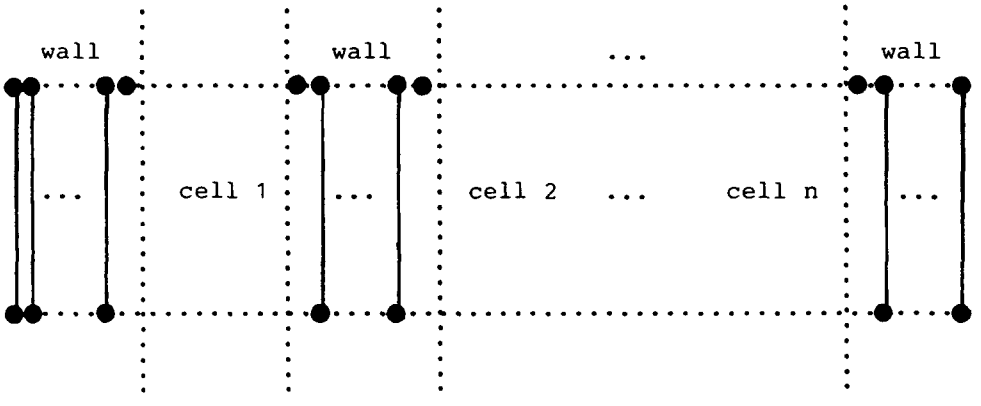

Fig. 5. Walls and cells.

Between each consecutive pair of walls we place a "cell". A cell has two kinds of edges, as described below.

For $1 \leqslant i \leqslant n$, the last upper vertex of the ith wall shares a common, lower neighbor with the first upper vertex in the next wall. The common neighbor together with its incident edges is called the "blocker" of the ith cell. This is illustrated in Fig. 6.

Note that the total number of blocker edges (taken over all cells) is $2n$.

Further, each cell contains "covering" edges. The covering edges for the ith cell represent incidences of edges in H with the vertex v_i corresponding to the cell. This representation is constituted as follows. Suppose that the edge list of vertex v_i consists of the $d(i)$ edges $f_{i1}, f_{i2}, \ldots, f_{id(i)}$. Then, for $1 \leqslant i \leqslant n$, the ith cell contains upper vertices $u_{i1}, u_{i2}, \ldots, u_{id(i)}$ corresponding to these incidences. These vertices are placed between the upper vertices of the ith and $(i+1)$th walls; specifically,

$$x(u_{ij}) = i(p+2) + \sum_{k=1}^{i-1} d(k) + j, \quad \text{where } 1 \leqslant j \leqslant d(i).$$

For each edge $f=(v_i, v_j) \in F$, we create a lower vertex w_f. Suppose that f is the rth edge on the edge list of v_i and the sth edge on the edge list of v_j (i.e. $f=f_{ir}=f_{js}$). Then we join w_f to u_{ir} and to u_{js} by what we call "covering" edges. These are illustrated in Fig. 7. Note that the total number of covering edges (taken over all cells) is $2m$. For the sake of completeness, note that the inventory of lower vertices is as follows. There are $(n+1)p$ lower vertices incident to wall edges, there are n lower vertices in cell blockers, and there are m lower vertices incident to covering edges.

Finally, we choose $B = p(n+1) + 2n - K + m$, completing the construction of the OPF instance. Note that this transformation can be done in time polynomial in the size of the description of the VC instance.

Next we show that our transformation maps "yes" instances of VC to "yes" instances of OPF.

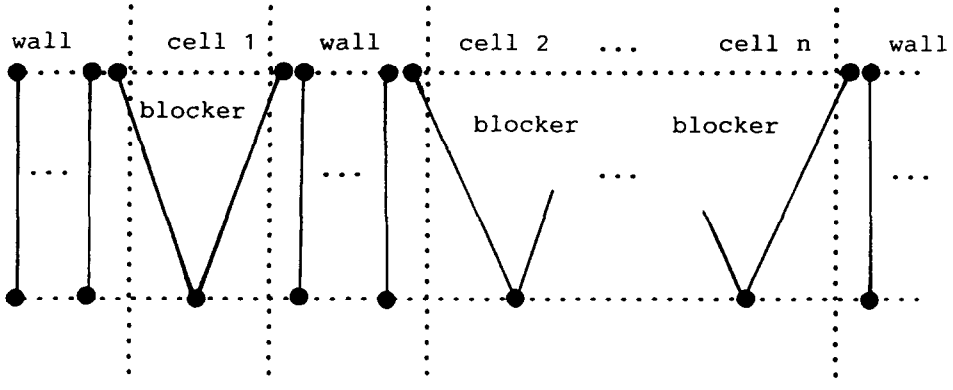

Fig. 6. Walls, cells, and blockers.

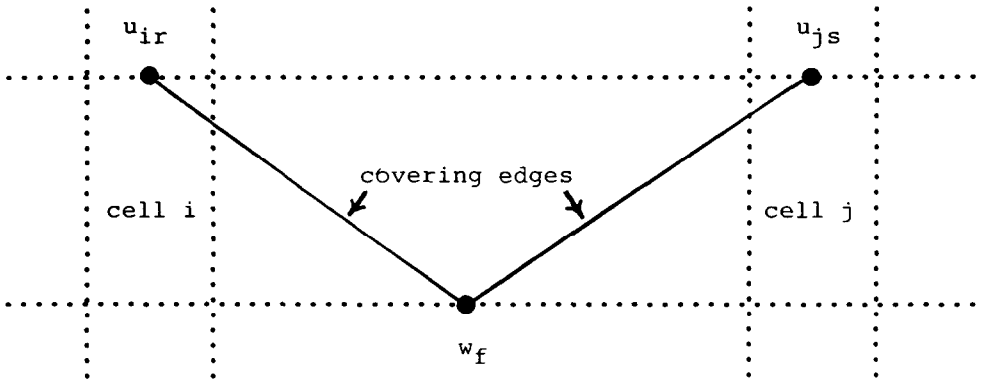

Fig. 7. Covering edges.

Suppose that H has a vertex cover V' such that $|V'| \leqslant K$. Then we construct a biplanar subgraph G' of the transformed graph G as follows.

We include in G' all $p(n+1)$ wall edges.

For $1 \leqslant i \leqslant n$, the choice of ith cell blocker edges depends on whether v_i is in the vertex cover V'. If $v_i \in V - V'$, then we include both edges of the blocker in G'. If $v_i \in V'$, then we include the left blocker edge only. This adds two blocker edges if $v_i \in V - V'$, and one blocker edge if $v_i \in V'$. Hence G' contains at least $2(n-K)+K = 2n-K$ blocker edges.

For each edge $f \in F$, we can take one covering edge incident to w_f without destroying the biplanarity of G'. To do this, we choose an endpoint v_i of edge f that is in V' (if both endpoints of f belong to V', then we choose v_i arbitrarily from them). Suppose that f is the jth edge on the edge list of v_i. Since there is only one blocker edge in the ith cell, we can include the covering edge (w_f, u_{ij}) in G' without destroying biplanarity. This puts a total of m covering edges into G'.

Thus the total of edges in G' is at least $p(n+1)+2n-K+m = B$.

Next we show that only "yes" instances of VC can transform to "yes" instances of OPF. Suppose that we have a biplanar drawing of a subgraph G' with at least B edges, and suppose that the biplanar drawing respects the prescribed positions for the vertices of U. We assume that G' is maximal in the sense that no more edges of G can be added to G' while retaining biplanarity. The number of edges of covering, wall and blocker type may vary from one such maximal subgraph to another. We assume that no other maximal subgraph contains more edges of the covering type than does G' (other subgraphs may have a larger total number of edges).

Let w be the number of walls of G that contribute all their edges to G'; then the total number of edges in G' would be at most $pw+2m+2n$, even if all the blocker and covering edges were included in G'. Thus

$$pw+2m+2n \geqslant B = p(n+1)+2n-K+m,$$

so

$$p(n+1-w) \leqslant K+m.$$

It follows by the choice of $p=2(m+n)+1$ specified in the transformation that $w=n+1$; i.e. every wall edge of G is in G'. Without loss of generality, assume that the lower vertices of wall edges of a given wall appear consecutively in the drawing of G'. We call those lower vertices that appear after the lower wall vertices of one wall, but before the lower wall vertices of the next wall, the *lower part* of the cell that lies between the two walls.

The fact that each wall edge of G belongs to G' has two simple but important consequences. First, the number of remaining edges of G', i.e. the number of blocker and covering edges in G', must be at least

$$B-p(n+1)=2n-K+m.$$

Second, for each edge (u, v) in G', if u is in the upper part of the ith cell or if u is the first or last upper vertex of the next or previous wall, then v must be positioned in the lower part of the ith cell in the biplanar drawing of G'. We say that the ith cell "contains" (u, v).

Let V' be the set of vertices $v_i \in V$ such that there is at least one covering edge contained in the ith cell. We prove that V' is a vertex cover for H of size at most K. First, we verify that its size is at most K.

The biplanarity of the drawing of G' implies that if the ith cell contains both edges of its blocker then it cannot have any covering edges and so v_i is not in V'. However, a cell with one blocker edge may have covering edges. Thus the number of blocker edges in G' is at most

$$2(n-|V'|)+|V'|=2n-|V'|.$$

Further, each lower vertex w_f corresponding to an edge $f \in F$ can contribute at most one of its edges to G'; hence there are at most m covering edges.

Thus the total number of nonwall edges in G' is at most $2n-|V'|+m$. Hence $2n-|V'|+m \geqslant 2n-K+m$, and so $|V'| \leqslant K$.

Finally, we claim that V' is a vertex cover for H. Suppose, to the contrary, that $f=(v_i, v_j) \in F$ has neither endpoint in V'. We can assume without loss of generality that in the drawing of G', w_f appears as a vertex of degree 0 positioned in the lower part of the ith cell. Suppose f is the rth edge in the edge list of vertex v_i. If any covering edges appeared in the ith cell in G', then we could add (w_f, v_{ir}) to the drawing of G' without destroying biplanarity, repositioning upper vertices or reducing the number of covering edges. This would contradict the maximality of G'. Thus the ith cell contains no covering edges and therefore contains two blocker edges. We can replace the contents of this cell by one blocker edge and the covering edge (w_f, v_{ir}) without decreasing the number of edges of G' and without destroying biplanarity or repositioning upper vertices. This contradicts the maximality of G' with respect to covering edges. □

OPF is NP-complete under various restrictions of the input. The proof above shows that it is NP-complete even when the degree of every vertex in the upper part is at most one and the degree of every vertex in the lower part is at most two.

4. All positions fixed

Consider the following problem: given a bipartite graph $G=(U, L, E)$, a position $x(u)$ on λ_U for each $u \in U$, and a position $x(v)$ on λ_L for each $v \in L$, find a subgraph H of G such that H has a biplanar drawing with its vertices at the prescribed positions and such that the number of edges of H is as large as possible.

We show that an efficient algorithmic solution for this problem can be obtained by first transforming it to the problem of finding a longest ascending subsequence C of a given sequence σ of integers, and then using an algorithm such as that in [19] that solves the longest ascending subsequence problem.

The transformation to the longest ascending subsequence problem requires two order relations on the edges of G: one on their upper endpoints, one on their lower endpoints. These relations are as follows. If (s, t) and (u, v) are edges with $s, u \in U$ and $t, v \in L$, then $(s, t) <_U (u, v)$ if $x(s) < x(u)$ and $(s, t) <_L (u, v)$ if $x(t) < x(v)$.

Let σ be the sequence of all the edges of G sorted first on $<_L$ and then on $<_U$ to break ties. An example is in Fig. 8. The following lemma describes the relationship between biplanarity and σ.

Lemma 4.1. *A set of edges of G is biplanar if and only if it forms a subsequence of σ that is nondecreasing under $<_U$.*

Proof. If the edge (s, t) precedes the edge (u, v) in a subsequence of σ that is nondecreasing under $<_U$, then $(s, t) <_L (u, v)$ and $(s, t) <_U (u, v)$. Thus (s, t) is entirely to the left of (u, v), and the two edges cannot cross. The other direction of the proof is clear. □

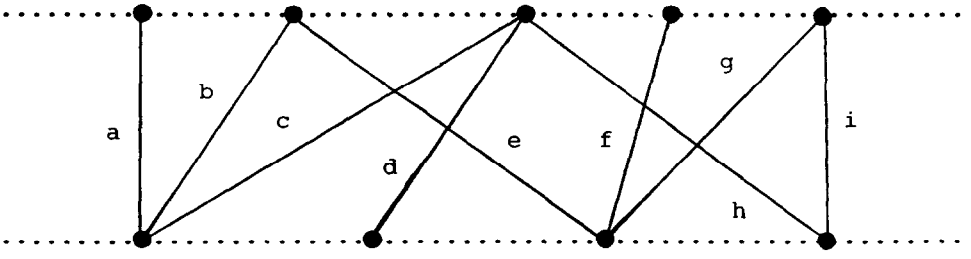

Fig. 8. A graph and its sequence σ.

Any algorithm for the longest ascending subsequence problem [3, 19] can be applied to find a longest subsequence of σ that is nondecreasing under $<_U$. This subsequence forms a maximum cardinality biplanar subgraph of G.

The best upper bound for the time required to solve the longest ascending subsequence problem is $O(m \log r)$, where m is the size of σ and r is the size of the output [19]. The initial construction of σ may take time $O(n \log n)$, where $n = |U| + |L|$. Note that in most cases of interest, $r < n \leqslant m$.

The following theorem summarizes these results.

Theorem 4.2. *A maximum biplanar subgraph C of $G = (U, L, E)$ with the positions of all vertices fixed can be found in time $O(m \log r + n \log n)$, where $m = |E|$, $r = |C|$, and $n = |U| + |L|$.*

In some applications, σ can be obtained in linear time. For instance, the vertices may be supplied in order of their x coordinates. In this case, σ may be constructed as follows. Suppose that $L = \{v_i: 1 \leqslant i \leqslant |L|\}$, where $x(v_1) \leqslant x(v_2) \leqslant \cdots \leqslant x(v_{|L|})$, and denote by E_i the sequence of edges incident with v_i, sorted on $<_U$. The concatenation of the sequences $E_1, E_2, \ldots, E_{|L|}$ is σ. This reduces the total complexity to $O(m \log r + n)$.

The problem of this section can also be transformed to the problem of finding a "maximum chain" in a set of points. Maximum chain problems and variants have been studied, for example, in [2, 16]. However, the transformation to the longest ascending subsequence problem seems to provide a particularly simple, direct solution to the problem of this section.

Acknowledgment

We would like to thank Dr. M. Orlowski for references to the longest ascending subsequence problem.

References

[1] D. Angluin and L.G. Valiant, Fast probabilistic algorithms for Hamilton circuits and matchings, *J. Comput. System Sci.* **18** (1979) 155–193.

[2] M.J. Atallah and S.R. Kosaraju, An efficient algorithm for maxdominance with applications. *Algorithmica* **4** (1989) 221–236.

[3] E.W. Dijkstra, Some beautiful arguments using mathematical induction, *Acta Inform.* **13** (1980) 1–8.

[4] P. Eades and D. Kelly, Heuristics for reducing crossings in 2-layered networks, *Ars Combin.* **21A** (1986) 89–98.

[5] P. Eades and X. Lin, How to draw a directed graph, in: *Proc. 1989 IEEE Workshop on Visual Languages* (IEEE Computer Soc. Press, Silver Spring, MD, 1989) 13–17.

[6] P. Eades, B.D. McKay and N.C. Wormald, On an edge crossing problem, in: *Proc. 9th Australian Computer Science Conf.* (Australian National University, 1986) 327–334.

[7] P. Eades and K. Sugiyama, How to draw a directed graph, *J. Inform. Process.* **13** (1990) 424–437.

[8] P. Eades and R. Tamassia, Algorithms for drawing graphs: an annotated bibliography, Tech. Report CS-89-09, Department of Computer Science, Brown University, 1989.
[9] P. Eades and N.C. Wormald, Edge crossings in drawings of bipartite graphs, Tech. Report 108, Department of Computer Science, University of Queensland, 1989.
[10] E.R. Gansner, S.C. North and K.P. Vo, DAG – a program that draws directed graphs, *Software Practice Experience* **18** (1988) 1047–1062.
[11] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of* NP-*completeness* (Freeman, New York, 1979).
[12] M.R. Garey and D.S. Johnson, Crossing number is NP-complete, *SIAM J. Algebraic Discrete Methods* **4** (1983) 312–316.
[13] D.J. Gschwind and T.P. Murtagh, A recursive algorithm for drawing hierarchical directed graphs, Tech. Report CS-89-02, Williams College, Williamstown, MA, 1989.
[14] D. Karger, R. Motwani and G.D.S. Ramkumar, On approximating the longest path in a graph, in: F. Dehne, J.-R. Sack, N. Santoro and S. Whitesides, eds., *Algorithms and Data Structures (Proc. 3rd Workshop on Algorithms and Data Structures WADS'93)*, Lecture Notes in Computer Science, Vol. 709 (Springer, Berlin, 1993) 421–432.
[15] D. Kelly, A view to graph layout problems, Masters Thesis, Department of Computer Science, University of Queensland, 1987.
[16] R.D. Lou, M. Sarrafzadeh and D.T. Lee, An optimal algorithm for the maximum two-chain problem, in: *SODA 90, Proc. 1st Ann. ACM–SIAM Symp. on Discrete Algorithms*, San Francisco (1990) 149–158.
[17] E. Makinen, Experiments in drawing 2-level hierarchical graphs, Tech. Report A-1988-1, Department of Computer Science, University of Tampere, 1988.
[18] E. Messinger, Automatic layout of large directed graphs, Tech. Report 87-07-08, Department of Computer Science, University of Washington, 1987.
[19] M. Orlowski and M. Pachter, An algorithm for the determination of a longest increasing subsequence in a sequence, *Comput. Math. Appl.* **17** (1989) 1073–1075.
[20] F.N. Paulich and W.F. Tichy, EDGE: an extendible directed graph editor, *Software Practice Experience* **20** (1990) 63–88.
[21] L.A. Rowe, M. Davis, E. Messinger, C. Meyer, C. Spirakis and A. Tuan, A browser for directed graphs, *Software Practice Experience* **17** (1987) 61–76.
[22] C. Sechen, *VLSI Placement and Global Routing Using Simulated Annealing* (Kluwer, Dordrecht, 1988).
[23] K. Sugiyama, A cognitive approach for graph drawing, *Cybernet. Systems* **18** (1987) 447–488.
[24] K. Sugiyama and K. Misue, Visualizing structural information: hierarchical drawing of a compound digraph, Tech. Report 86, IIAS-SIS, Fujitsu Limited, Numazu Shizuoka Japan, 1989.
[25] K. Sugiyama, S. Tagawa and M. Toda, Methods for visual understanding of hierarchical system structures, *IEEE Trans. Systems Man Cybernet.* **SMC-11** (1981) 109–125.
[26] J.D. Ullman, *Computational Aspects of VLSI* (Computer Science Press, Rockville, MD, 1984).