Discrete Comput Geom 1:343-353 (1986)

Discrete & Computational
Geometry

Rectilinear Planar Layouts and Bipolar Orientations of Planar Graphs

Pierre Rosenstiehl[1]* and Robert E. Tarjan[2]

[1] Centre de Mathematique Sociale, Paris, France

[2] Computer Science Department, Princeton University, Princeton, NJ 08544, USA, and AT&T Bell Laboratories, Murray Hill, NJ 07974, USA

Abstract. We propose a linear-time algorithm for generating a planar layout of a planar graph. Each vertex is represented by a horizontal line segment and each edge by a vertical line segment. All endpoints of the segments have integer coordinates. The total space occupied by the layout is at most n by at most $2n-4$. Our algorithm, a variant of one by Otten and van Wijk, generally produces a more compact layout than theirs and allows the dual of the graph to be laid out in an interlocking way. The algorithm is based on the concept of a *bipolar orientation.* We discuss relationships among the bipolar orientations of a planar graph.

1. Introduction

Let $G=(V, E)$ be a planar graph consisting of a vertex set V and an edge set E. We shall denote the number of vertices by n and the number of edges by m. We assume $n \geq 3$; thus by Euler's formula $m \leq 3n-6$. We shall consider the problem of constructing a layout of G in the plane with no crossing edges. Solutions to this problem have potential applications in VLSI design [16], schematics [10], algorithm animation [2], and other areas.

Possible solutions to the layout problem depend upon the constraints imposed on the layout. The classical version of the problem asks for a *straight-line embedding*, i.e., a layout in which the vertices are mapped to points and the edges are mapped to straight-line segments, with the endpoints of each line segment being the images of the end vertices of the corresponding edge. Fáry [8] showed

* Research partly supported by the Agence de L'Informatique du Ministere de L'Industrie, France, under contract No. 83-285.

that every planar graph has a straight-line embedding, although he did not consider algorithms for finding one. Tutte [23] showed that every triconnected graph has a straight-line embedding in which all interior faces are convex; moreover, he devised a polynomial-time algorithm for finding such an embedding. Chiba *et al.* [3] proposed a linear-time algorithm for finding an embedding.

These algorithms have two drawbacks: they manipulate real numbers of high precision and they tend to bunch the vertices together, producing unsatisfactory layouts. (It is unknown whether every planar graph has a straight-line embedding such that the vertices map to integer lattice points with coordinates bounded by n^k for some constant k.) We can eliminate these drawbacks by relaxing the layout requirements. We require that each vertex be mapped to an integer lattice point and that each edge be mapped to a sequence of line segments connecting integer lattice points. Shiloach [19] showed that any planar graph has such a layout and Woods [24] provided an $O(n^2)$-time algorithm for finding one. The layouts of Shiloach and Woods occupy space $O(n)$ by $O(n)$, which is best possible in the worst case.

This approach has the drawback that it often produces layouts in which edges have many bends. Several authors independently proposed a layout regime that eliminates this drawback: each vertex is stretched by mapping it to a horizontal line segment rather than just a point, and each edge is mapped to a sequence of horizontal and vertical line segments. Cori and Hardouin-Duparc [4] proposed a layout algorithm in which each edge has at most four bends. They give no time bound for their algorithm. Greene [12] proposed an $O(n)$-time algorithm in which each edge has at most two bends. His algorithm has the nice feature that it can be used to produce an interlocking layout of the dual graph.

The ultimate version of this layout regime is to represent each edge by a single vertical line segment. Otten and van Wijk [17] showed that every planar graph has such a layout and they provided an $O(n)$-time algorithm to construct one. Independently, Duchet *et al.* [5] proved the existence of a layout for any planar graph and Fraysseix and Rosenstiehl gave an algorithm for finding a layout based on their left–right planar embedding algorithm [10], [11].

We propose a variant of the Otten–van Wijk method that can produce more compact layouts and allows the construction of an interlocking layout of the dual graph. Essentially the same algorithm has been proposed independently by Tamassia and Tollis [20]. Our algorithm is based on the concept of a *bipolar orientation*, which is related to the *st*-numbering notion used by Otten and van Wijk. Section 2 contains a description of our algorithm. The exact layout produced depends upon the choice of bipolar orientation; in general, there are many such orientations. In Section 3 we explore relationships among the different bipolar orientations of a planar graph. In Section 4 we conclude with some remarks about the problem of minimizing the layout area.

2. A Planar Layout Algorithm

Let G be a not-necessarily-planar undirected graph with two distinguished vertices, s and t. A *bipolar orientation* of G is a directed acyclic graph formed

from G by directing each undirected edge $\{v, w\}$ either from v to w or from w to v, such that s is the unique source (vertex with no incoming edges) and t is the unique sink (vertex with no outgoing edges). An equivalent notion, that of an *st*-numbering, was devised by Lempel *et al.* [15] for use in their efficient planarity-testing algorithm. An *st-numbering* of G is a numbering of the vertices from 1 through n such that s is numbered 1, t is numbered n, and every other vertex is adjacent both to a lower-numbered and to a higher-numbered vertex. Given an *st*-numbering, we can obtain a bipolar orientation by directing every edge from its lower-numbered to its higher-numbered end vertex. Given a bipolar orientation, we can obtain an *st*-numbering by numbering the vertices from 1 through n in topological order (in an order such that, if (v, w) is an edge directed from v to w, v is numbered less than w). Since topological orderings are computable in $O(n+m)$ time [14], [21], these notions are linear-time equivalent. A graph G with distinguished vertices s and t has a bipolar orientation if and only if $G^+ = (V, E \cup \{\{s, t\}\})$ is biconnected [15]. Such an orientation can be found in $O(n+m)$ time using depth-first search [6], [7], [22].

We need one more notion unrelated to planarity. Let $D = (V, E)$ be a directed acyclic graph with a unique source s. We define the *level* of a vertex v, denoted by $level(v)$, to be the maximum number of edges on a path from s to v. The levels of all vertices can be computed in linear time using the recurrence $level(s) = 0$, $level(w) = \max\{level(v)+1 | (v, w) \in E\}$ for $w \neq s$.

Now suppose we are given the topology of a planar embedding of a connected planar graph G, in the form of a circular list, for each vertex, of the incident edges appearing clockwise around the vertex. We call these circular lists *rotations*; a planar graph together with a set of rotations is a *plane graph*. A set of rotations can be constructed in $O(n)$ time using any of the known linear-time planarity-testing algorithms [1], [11], [13], [15]. Given a set of rotations, we can construct the boundaries of the faces of the embedding in $O(n)$ time. To do this, we select an edge $\{v, w\}$, traverse it from v to w, traverse the nearest edge clockwise from $\{v, w\}$ around w, and continue in this way until returning to $\{v, w\}$ from v. This gives us the first facial boundary, which has been traversed in a counterclockwise direction unless it is the exterior face. We repeat this traversal process to obtain all the facial boundaries, always starting with an edge in a direction in which it has not yet been traversed. Each edge appears exactly twice on facial boundaries, once in each direction. With respect to a directed edge (v, w), the *left face* is the face whose boundary contains (v, w), and the *right face* is the face whose boundary contains (w, v). The *dual graph* G' of G consists of the faces of G as vertices, with two faces adjacent if they are the left and right faces of some edge. We can construct G' in $O(n)$ time.

We assume G is biconnected. If not, we can make G biconnected in $O(n)$ time while preserving planarity by adding appropriate dummy edges across the faces [24]. Once the layout is computed we can ignore the layout of the dummy edges.

The layout algorithm consists of the following four steps (see Figs. 1 and 2):

Step 1. Compute the facial boundaries, the left and right faces of each edge, and the dual graph G' of G.

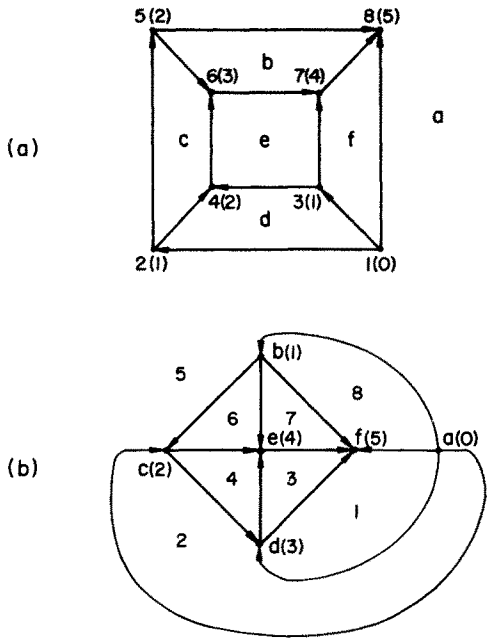

Fig. 1. A planar graph and its dual. (a) A planar graph with a bipolar orientation. The vertices are numbered in an *st*-order with levels in parentheses. (b) The dual of the graph with the imposed bipolar orientation. The faces are lettered in an *st*-order with levels in parentheses.

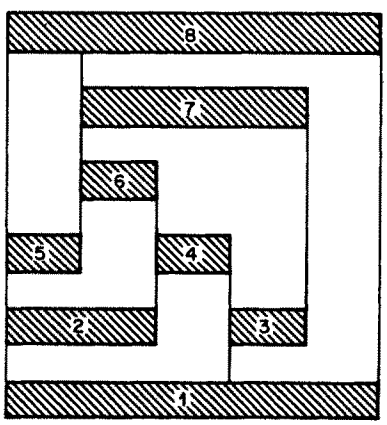

Fig. 2. The layout of the graph in Fig. 1(a) implied by the selected bipolar orientation. The vertices have been given positive thickness for clarity.

Step 2. Select an arbitrary edge $\{s, t\}$ of G and direct the edges of G to form a bipolar orientation D of G with source s and sink t. Compute the level of every vertex in D.

Step 3. Direct the edges of the dual grapah G' as follows: if f_L and f_R are, respectively, the left and right faces of some edge (v, w) of D, direct the dual edge from f_L to f_R if $(v, w) \neq (s, t)$ and from f_R to f_L if $(v, w) = (s, t)$. The result is a bipolar orientation D' of G'; the source and sink of D' are the right and left faces of (s, t), respectively. Compute the level of every face (as a vertex in D').

Step 4. Lay out G by mapping each edge (v, w) of D with left face f into the vertical line segment with endpoints $(level(f), level(v))$ and $(level(f), level(w))$ and mapping each vertex v into the horizontal line segment with endpoints $(x_1, level(v))$ and $(x_2, level(v))$, where $x_1 = \min\{level(f) \mid f$ is the left face of some edge (v, w) or (w, v) of $D\}$ and $x_2 = \max\{level(f) \mid f$ is the left face of some edge (v, w) or (w, v) of $D\}$.

Note. A vertex of degree two distinct from s and t is laid out as a line segment consisting of a single point. □

The correctness of the layout algorithm follows from the correctness of the Lempel-Even-Cederbaum planarity-testing algorithm [15]. We shall sketch a proof.

Lemma 1 [20], [24]. *All the entering edges of any vertex in D appear consecutively in the rotation around v, as do all the exiting edges.*

Lemma 1 follows with a little work from the definition of a bipolar orientation. It implies that the layout respects the rotations, i.e., the line segments representing edges appear around the line segment representing a vertex in the same order as in the corresponding rotation. The dual to Lemma 1 is also true:

Lemma 2 [20]. *The boundary of every face consists of exactly two directed paths in D.*

Lemma 2 also follows from the definition of a bipolar orientation. We define the *right boundary* of a face to be the set of edges for which the face is the left face, and the *left boundary* to be the set of edges for which the face is the right face. By Lemma 2, the left and right boundaries of a face are single paths in D. Lemma 2 and the layout definition imply that the right boundary of each face is laid out as a single vertical line segment.

Lemma 3 [20]. *The dual directed graph D' is acyclic and bipolar, with source the right face of (s, t) and sink the left face of (s, t).*

Lemma 3 follows from Lemma 2.

Theorem 1. *Any two line segments generated by the layout algorithm are disjoint except that the endpoints of a line segment representing an edge are on the line segments representing the pair of vertices making up the edge. The exterior face of the layout is the source f_0 of D'.*

Proof. Suppose we lay out the faces of G other than f_0 one at a time in a topological order with respect to D'. As the faces are laid out, there is a moving *frontier* that consists of a path in D from s to t, corresponding to a sequence of horizontal and vertical line segments forming the rightmost extent of the partial layout. The initial frontier r_0 is the right boundary of f_0 and is laid out as a line segment with y-coordinate zero. If r_i is the current frontier and f_{i+1} is the next face to be laid out, the left boundary of f_{i+1} is part of r_i, and the new frontier r_{i+1} is r_i with the left boundary of f_{i+1} replaced by the right boundary of f_{i+1}. The new face f_{i+1} is laid out to the right of the line segments representing r_i, and the layout definition implies that the new line segments representing the right boundary of f_{i+1} cross no previously laid out line segments, and intersect only the horizontal line segments representing the minimum and maximum vertices of f_{i+1}. The theorem follows by induction on the number of faces laid out. □

The layout algorithm runs in $O(n)$ time and constructs a layout that is of height at most n and width at most $2n-4$. (This is the maximum number of faces.) The Otten-van Wijk algorithm has the same worst-case bounds but always uses height n, as it embeds every vertex at a different height. By redefining the level calculations appropriately, we can modify the algorithm so that the line segments representing the vertices and edges have specified thicknesses.

The vertex and face levels also define a layout of the dual graph. If we subtract $\frac{1}{2}$ from all x-coordinates and add $\frac{1}{2}$ to all y-coordinates of the dual, we obtain a layout that interlocks correctly with the layout of the original graph, with one anomaly. The only intersections are of each edge with its dual, except that (s, t) does not intersect with its dual; instead, the line segment representing t intersects

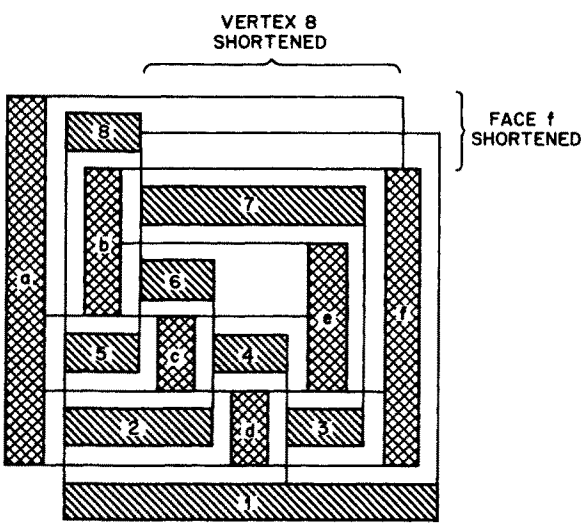

Fig. 3. Interlocked layout of the graph in Fig. 1(a) and its dual. Vertex 8 and face f have been shortened to eliminate their intersection.

the segment representing the left face of (s, t). If we shorten these segments and extend the segments representing (s, t) and its dual by a horizontal and a vertical segment, respectively, we obtain a layout of the graph and its dual in which the only intersections are of each edge with its dual (see Fig. 3).

3. Bipolar Orientations in Planar Graphs

Since the layout produced by the algorithm of Section 2 depends on the bipolar orientation chosen, it is natural to investigate the class of all such orientations. In this section we derive three results about bipolar orientations of planar graphs. A preliminary version of the first two of these results appeared as [18].

Our first result shows that Lemmas 1 and 2 characterize bipolar orientations. Let G be a biconnected plane graph, i.e., a planar graph with a specified embedding topology given in the form of rotations. An *angle* is a pair of edges $\{u, v\}, \{u, w\}$ such that $\{u, w\}$ occurs just after $\{u, v\}$ in the rotation around u. The *angle graph* A of G is the graph whose nodes are the vertices and faces of G and whose edges are the pairs $\{u, f\}$ such that u is a vertex on the boundary of face f. The edges of A correspond to the angles of G: an angle $\{u, v\}, \{u, w\}$ corresponds to the edge joining u with the face having (v, u) and (u, w) on its boundary (when the facial boundaries are generated as described in Section 2). Hence we shall sometimes refer to the edges of A as angles. The graph A is bipartite, planar, and has only four-sided faces (see Fig. 4.)

Let $\{s, t\}$ be a fixed edge of G. A *bipolar marking* is a marking of the edges of A with the colors 0 and 1 such that:

(i) all edges of A incident to s and t are marked 0;

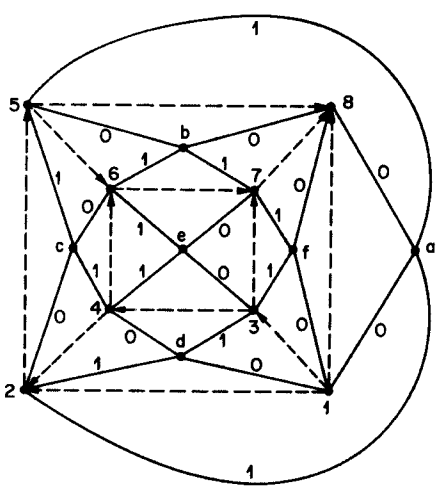

Fig. 4. The angle graph of the graph in Fig. 1(a), with the original graph indicated by dashed lines. A bipolar orientation and the corresponding bipolar marking are shown.

(ii) for any vertex of G other than s and t, exactly two incident edges in A are marked 1; and
(iii) for any face f of G, exactly two incident edges in A are marked 0.

Theorem 2. *The bipolar orientations of G with source s and sink t are in one-to-one correspondence with the bipolar markings of A.*

Proof. Consider a bipolar orientation D of G. Assign mark 1 to each angle $\{u, v\}, \{u, w\}$ such that exactly one of $\{u, v\}$ and $\{u, w\}$ is directed into u in D, and assign mark 0 to every other angle. Lemmas 1 and 2 imply that the result is a bipolar marking.

The proof of the converse uses the same face-by-face addition process as the proof of Theorem 1. Consider a bipolar marking of A. We define a moving frontier and incrementally orient the edges of G as follows. Let f_0 be the left face of (t, s). The initial frontier r_0 is the boundary of f_0 excluding the edge (t, s). The frontier r_0 is a path from s to t; orient its edges in D in the direction along the path. Given the current frontier r_i, which defines a path from s to t in G, select a face f_{i+1} as follows. Let u be the closest vertex to t along r_i such that the angle $\{u, v\}, \{u, w\}$ with v following u along r_i is marked 0. (There is such a vertex u since s is a candidate by (i).) Let x be the closest vertex to u on the part of r_i from u to t such that the angle $\{x, y,\}, \{x, z\}$ with z preceding x along r_i is marked zero. (There is such a vertex x since t is a candidate by (i).) It follows from (ii) and (iii) that there is a face f_{i+1} whose boundary contains (y, x), the reverse of the part of r_i from u to x, and (u, w). That is, the boundary of f_{i+1} consists of a path p from x to u that is the reverse of part of r_i, and a path q from u to x. In D, direct the edges on q along the direction of q, and form r_{i+1} from r_i by replacing the reverse of p by q. Repeat the process of redefining the frontier until it is (s, t).

An induction using the connectivity of the dual graph G' of G shows that every edge is assigned a direction by this process; once a face of G is selected, all adjacent faces must eventually be selected as well. The resulting orientation of G is obviously bipolar. □

Our second result shows how to obtain any bipolar marking from a given one. Suppose we have a bipolar marking of A. An *alternating cycle* is a cycle in A whose edge marks alternate between 0 and 1.

Theorem 3. *Let M_0 be any bipolar marking of A. Then any other bipolar marking of A (with respect to the same source-sink pair $\{s, t\}$) can be obtained from M_0 by flipping the marks of all edges on some edge-disjoint set of alternating cycles, and any such flipping produces a bipolar marking.*

Proof. Let M_1 be any other bipolar marking of A. The set of edges marked differently in M_0 and M_1 must contain an even number of edges incident with each vertex, and thus can be partitioned into a collection of edge-disjoint cycles. This implies the first part of the lemma. Flipping the marks of all edges on any

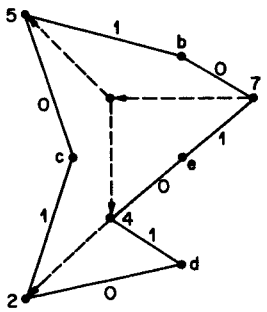

Fig. 5. The effect on the graph in Fig. 4 of flipping marks along the alternating cycle 2, *c*, 5, *b*, 7, *e*, 4, *d*. Only the cycle and the edges in G that change orientation are shown.

collection of edge-disjoint alternating cycles preserves the number of 0 and 1 edges incident to each vertex and thus preserves (i), (ii), and (iii). This implies the second part. □

Our third and last result characterizes the effect on the bipolar orientation of flipping the marks of edges on a single alternating cycle. Let $G \cup A$ denote the planar graph formed by adding the edges of G to A. $G \cup A$ has a planar embedding corresponding to the embedding of G. Let M_0 be a bipolar marking of A, corresponding to a bipolar orientation D_0 of G. Let c be a simple alternating cycle of M_0. Let M_1 be the marking formed from M_0 by flipping the marks of the edges on c, and let D_1 be the corresponding orientation of G.

Theorem 4. *Both s and t are on the same side of c in the planar embedding of $G \cup A$. D_1 is obtained from D_0 by reversing the orientation of all edges of G on the side of c not containing s and t (see Fig. 5).*

Proof. Let us call two edges of G *adjacent* if they form an angle. Any two edges of G on the same side of c in $G \cup A$ are connected by a path of adjacencies, none of which correspond to the angles forming c. It follows that the edges in G on one side of c either all have the same orientation in D_0 and D_1 or all have opposite orientations in D_0 and D_1. The theorem follows. □

4. Remarks

A natural goal in layout problems is to minimize the area of the layout. For the layout algorithm of Section 2, we would like to be able to find a bipolar orientation that minimizes the layout area. The results of Section 3 provide a brute-force approach to this problem: we find one bipolar orientation, generate all the others by finding alternating cycles, and compute the layout area for each. Unfortunately, there may be an exponential number of possible bipolar orientations to try. We conjecture that the problem of finding a bipolar orientation that minimizes the

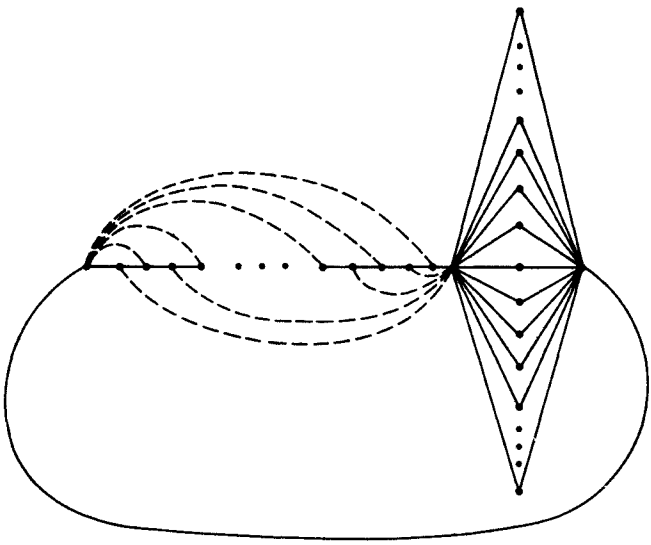

Fig. 6. A graph with an $\Omega(n)$ by $\Omega(n)$ layout regardless of what bipolar orientation is chosen. If the dashed edges are added, there is a bipolar orientation that produces an $O(n)$ by $O(1)$ layout.

layout area is NP-hard. (For various other layout regimes the area-minimization problem is known to be NP-hard; see, for example, [24].)

A related question is whether our algorithm produces a good layout even if we know how to choose a good bipolar orientation. Unfortunately the answer is no. For the graph of Fig. 6, our algorithm will produce an $\Omega(n)$ by $\Omega(n)$ layout, regardless of what bipolar orientation is used. However, there is an $O(n)$ by $O(1)$ layout, which the algorithm will produce if we add the indicated dummy edges to the graph and choose the right bipolar orientation. In general, the problem of minimizing layout area seems to be very hard; only heuristics with no performance guarantees are known.

Another question is how to lay out nonplanar graphs. Our algorithm can be extended to handle nonplanar graphs, at the cost of introducing two dog-legs (four extra line segments) per edge crossing. The coding of a plane graph by a double occurrence sequence, as proposed independently by Greene [12] and Fraysseix [10], has been extended by Fraysseix [9] to generate rectilinear layouts of nonplanar graphs in time linear in the graph size, using at most three line segments to represent each edge.

References

1. K. S. Booth and G. S. Lueker, Testing for the consecutive ones property, interval graphs, and graph planarity using *PQ*-tree algorithms, *J. Comput. System Sci.* **13** (1976), 335–379.

2. M. H. Brown and R. Sedgewick, A system for algorithm animation, Technical Report No. CS-84-01, Department of Computer Science, Brown University, Providence, RI, 1984.
3. N. Chiba, T. Yamanouchi, and T. Nishizeki, Linear algorithms for convex drawings of planar graphs, *Proceedings of the Silver Jubilee Conference on Combinatorics, 1982*, University of Waterloo, Waterloo, Ontario, to appear.
4. R. Cori and J. Hardouin-Duparc, Manipulation des cartes planaires à partir de leur codage, Département de Mathématiques, Université de Bordeaux, Talence, France, 1975.
5. P. Duchet, Y. Hamidoune, M. Las Vergnas, and H. Meyniel, Representing a planar graph by vertical lines joining different intervals, *Discrete Math.* **46** (1983), 319–321.
6. J. Ebert, *st*-ordering the vertices of biconnected graphs, *Computing* **30** (1983), 19–33.
7. S. Even and R. E. Tarjan, Computing an *st*-numbering, *Theoret. Comput. Sci.* **2** (1976), 339–344.
8. I. Fáry, On straight line representing of planar graphs, *Acta. Sci. Math.* (Szeged) **11** (1948), 229–233.
9. H. de Fraysseix, Drawing planar and non-planar graphs from the half-edge code, to appear.
10. H. de Fraysseix and P. Rosenstiehl, Structures combinatoires pour des traćes automatiques de réseaux, *Proceedings of the Third European Conference on CAD/CAM and Computer Graphics*, 332–337, 1984.
11. H. de Fraysseix and P. Rosenstiehl, L'algorithme gauche-droite pour le plongement des graphes dans le plan, to appear.
12. D. H. Greene, Efficient coding and drawing of planar graphs, Xerox Palo Alto Research Center, Palo Alto, CA, 1983.
13. J. Hopcroft and R. Tarjan, Efficient planarity testing, *J. Comput. Mach.* **21** (1974), 549–568.
14. D. E. Knuth, *The Art of Computer Programming*, Vol. 1, 2nd ed., 258–265, Addison-Wesley, Reading, MA, 1973.
15. A. Lempel, S. Even, and I. Cederbaum, An algorithm for planarity testing of graphs, *Theory of Graphs* (International Symposium, Rome, July 1966) (P. Rosenstiehl, ed.), 215–232, Gordon and Breach, New York, 1967.
16. C. Mead and L. Conway, *Introduction to VLSI Systems*, Addison-Wesley, Reading, MA, 1980.
17. R. H. J. M. Otten and J. G. van Wijk, Graph representations in interactive layout design, *Proceedings of the IEEE International Symposium on Circuits and Systems* 914–918, 1978.
18. P. Rosenstiehl, Embedding in the plane with orientation constraints, *Ann. N.Y. Acad. Sci.*, to appear.
19. Y. Shiloach, Arrangements of planar graphs on the planar lattice, Department of Applied Mathematics, The Weizmann Institute of Science, Rehovot, Israel, 1976.
20. R. Tamassia and I. G. Tollis, Algorithms for visibility representations of planar graphs, Coordinated Science Laboratory, University of Illinois, Urbana, IL, 1985.
21. R. E. Tarjan, Finding dominators in directed graphs, *SIAM J. Comput.* **3** (1974), 62–69.
22. R. E. Tarjan, Two streamlined depth-first search algorithms, *Fund. Inform.*, **IX** (1986), 85–94.
23. W. T. Tutte, How to draw a graph, *Proc. London Math. Soc.* **13** (1963), 743–768.
24. D. R. Woods, Drawing planar graphs, Report No. STAN-CS-82-943, Computer Science Department, Stanford University, Stanford, CA, 1981.

Received August 21, 1985.