Discrete Comput Geom 7:381–401 (1992)

Discrete & Computational
Geometry

Area Requirement and Symmetry Display of Planar Upward Drawings*

Giuseppe Di Battista,[1] Roberto Tamassia,[2] and Ioannis G. Tollis[3]

[1] Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza," Via Salaria, 113 Roma, Italy 00198

[2] Department of Computer Science, Brown University, Providence, RI 02912-1910, USA

[3] Department of Computer Science, University of Texas at Dallas, Richardson, TX 75083-0688, USA

Abstract. In this paper we investigate the problem of constructing planar straight-line drawings of acyclic digraphs such that all the edges flow in the same direction, e.g., from bottom to top. Our contribution is twofold. First we show the existence of a family of planar acyclic digraphs that require exponential area for any such drawing. Second, motivated by the preceding lower bound, we relax the straight-line constraint and allow bends along the edges. We present a linear-time algorithm that produces drawings of planar *st*-graphs with a small number of bends, asymptotically optimal area, and such that symmetries and isomorphisms of the digraph are displayed. If the digraph has no transitive edges, then the drawing obtained has no bends. Also, a variation of the algorithm produces drawings with exact minimum area.

1. Introduction

A classical result shows that every planar graph admits a planar drawing with straight-line edges (*straight-line drawing*) [11], [35], [36], [46]. However, the existence of planar straight-line drawings with vertices placed at grid points (i.e., with integer coordinates) and polynomial area has been one of the most important

* Research supported in part by Cadre Technologies Inc., by the ESPRIT II Basic Research Actions Program of the EC under Contract No. 3075 (project ALCOM), by the National Science Foundation under Grant CCR-9007851, by the Office of Naval Research and the Defense Advanced Research Projects Agency under Contract N00014-83-K-0146 and ARPA order 6320, amendment 1, by the Progetto Finalizzato Sistemi Informatici e Calcolo Parallelo of the Italian National Research Council, by the Texas Advanced Research Program under Grant No. 3972, and by the U.S. Army Research Office under Grant DAAL03-91-G-0035.

and intriguing open problems in this field [33]. This question has been positively settled by de Fraysseix *et al.* [12], [13] and, independently, by Schnyder [34], who show that every n-vertex planar graph admits a planar straight-line drawing with vertices placed at grid points and $O(n^2)$ area.

In this paper we investigate the problem of constructing planar drawings of acyclic digraphs with the additional requirement that all edges flow in the same direction, e.g., from bottom to top. The construction of such *upward* drawings is very important for the display of hierarchic structures in data presentation applications. Digraphs that are customarily represented by upward drawings include PERT diagrams, ISA hierarchies, Hasse diagrams, and subroutine-call graphs. The construction of planar upward drawings can be viewed as computing "geometric realizations" of planar acyclic digraphs as monotone subdivisions. Notice that not all planar acyclic digraphs admit a planar upward drawing [6].

The contribution of this paper is twofold. First we show that there exists a family of planar acyclic digraphs that require exponential area in any planar straight-line upward drawing with vertices placed at grid points. Namely, we show that, for any positive integer n, there exists a planar acyclic digraph G_n with $2n + 2$ vertices such that any planar straight-line upward drawing of G_n with vertices placed at grid points has area $\Omega(2^n)$. This result sharply contrasts with the one of [12], [13], and [34]. Our lower bound is also valid in a very general drawing model, where the constraint on the vertices placed at grid points is relaxed by requiring only a minimum unit distance between any two vertices, or any other similar "finite-resolution" requirement.

Second, motivated by the preceding lower bound, we relax the constraint that edges must be drawn as straight lines, and consider the problem of drawing a planar acyclic digraph allowing *bends* along the edges (*polyline drawing*). In addition to the planar and upward requirements, we investigate the detection and display of symmetries and of isomorphic components. We present an $O(n)$-time algorithm for constructing a planar polyline upward drawing with at most $2n - 5$ bends, $O(n^2)$ area, and vertices placed at grid points. This algorithm is capable of displaying the symmetries of the digraph as well as its isomorphic subgraphs. The best previous algorithm for upward polyline drawings [6] uses $O(n^2)$ area and at most $(10n - 31)/3$ bends, without displaying symmetries or isomorphic subgraphs. The importance of the display of symmetries in the drawing of a graph has been pointed out by Lipton *et al.* [24], who give a model for measuring the symmetry of straight-line drawings. The problem of displaying symmetries was previously only partially solved for trees [27], [30], [38] and outerplanar graphs [26]. In general, it is NP-complete to detect whether a graph admits a symmetric drawing [25].

We also show that digraphs that do not contain transitive edges are drawn without bends and in such a way that the transitive closure is geometrically characterized by the dominance relation between the points associated with the vertices. Such drawings, called *dominance drawings*, display the two-dimensionality of the partial order defined by the digraph [21], [22].

Finally, a variation of the algorithm constructs dominance drawings with exact minimum area. Notice that the general area minimization problem for planar drawings is NP-hard [7].

The rest of this paper is organized as follows. Section 2 surveys previous work on algorithms for drawing planar graphs and digraphs. The exponential lower bound on the area of planar straight-line upward drawings is presented in Section 3. Drawing algorithms are given in Section 4. Section 5 concludes the paper with open problems.

2. Graph Drawing Algorithms

The problem of constructing readable drawings of graphs arises in several applications such as circuit schematics and diagrams for information systems analysis and design. Several references on graph drawing algorithms can be found in [9], [10], and [40].

Let Γ be a drawing of a graph G; Γ maps each vertex of G to a distinct point of the plane and each edge (u, v) of G to a simple Jordan curve with endpoints u and v. We say that Γ is a *polyline* drawing if each edge is a polygonal chain; Γ is a *straight-line* drawing if each edge is a straight-line segment; Γ is *planar* if no two edges intersect, except, possibly, at common endpoints.

The *area* of a drawing Γ can be defined in several ways: Regarding lower bounds, we define it as the area of the smallest polygon covering Γ. Regarding upper bounds, we define it more restrictively as the area of the smallest rectangle with sides parallel to the x and y axes covering Γ. The finite resolution of display and printing devices (and of the human eye) requires that some constraint be placed on the drawing so that its dimensions cannot be arbitrarily scaled down. Any constraint which implies a finite minimum area for the drawing of a graph is called a *resolution rule*. Two typical resolution rules are requiring integer coordinates for the vertices or a minimum distance δ between any two vertices. When a resolution rule is given, it is meaningful to consider the problem of finding drawings with minimum area. This problem is important in data-presentation applications and circuit layout.

Planarity is a fundamental aesthetic criterion of readability for the drawing of a graph, and the problem of constructing planar drawings of planar graphs has been extensively investigated. Every planar graph admits a planar straight-line drawing [11], [35], [36], [46]. However, the algorithms for constructing planar straight-line drawings given in [3], [29], and [45] use real arithmetic for the computation of the vertex coordinates, and if a resolution rule is given, then the area of the drawing appears to become exponential.

The important question whether there is some resolution rule such that every planar graph admits a planar straight-line drawing with polynomial area has been positively settled in [12], [13], and [34] where it is shown that every n-vertex planar graph admits a planar straight-line drawing with vertices placed at grid points and $O(n^2)$ area. Such drawings can be constructed in $O(n)$ time [4], [34]. If bends are allowed along edges, algorithms for constructing planar polyline drawings of undirected graphs with $O(n^2)$ area are presented in [7], [37], [39], [43], and [47]. A *visibility representation* of a planar graph maps each vertex v to a horizontal segment $\Gamma(v)$ and each edge (u, v) to a vertical segment $\Gamma(u, v)$ that

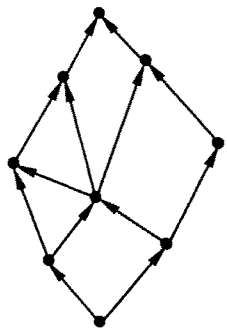

Fig. 1. A planar straight-line upward drawing

has endpoints on $\Gamma(u)$ and $\Gamma(v)$ and does not intersect any other vertex-segment. This representation can be constructed in $O(n)$ time [33], [42].

In addition to planarity, the display of symmetries and of isomorphic subgraphs are crucial features in graph drawing. In general, it is NP-complete to detect whether a graph admits a symmetric drawing [25]. A model for measuring the symmetry of a straight-line drawing of a graph is given in [24]. Algorithms for detecting and displaying symmetries in planar straight-line drawings of binary trees are presented in [30] and [38]. The problem of displaying symmetries in straight-line drawings of free trees and outerplanar graphs is investigated in [26] and [27]. An effective approach to symmetry display in (nonplanar) straight-line drawings is given in [8].

Now we consider directed graphs, and we say that a drawing of a digraph is *upward* if each edge is a curve monotonically increasing in the y-direction. An example of a straight-line upward drawing is shown in Fig. 1.

A characterization of the class of digraphs that admit a planar upward drawing has been given in [6] and [20]. This class consists of the subgraphs of *planar st-graphs* [23], which are planar acyclic digraphs with exactly one source (vertex without incoming edges), s, and one sink (vertex without outgoing edges), t, embedded in the plane with s and t on the boundary of the external face (see Fig. 1). Planar upward drawings with convex faces are investigated in [44].

Algorithms for constructing planar upward drawings are given in [6]. A first algorithm constructs, in $O(n \log n)$ time, straight-line drawings with vertices placed at real coordinates. Another algorithm constructs, in $O(n)$ time, polyline drawings with vertices placed at grid points, $O(n^2)$ area, and at most $(10n - 31)/3$ bends.

Covering digraphs of partial orders (order diagrams) are usually represented by straight-line upward drawings. A survey on drawing techniques for order diagrams appears in [31]. Characterizations of planar diagrams are presented in [20], [21], and [28].

3. An Exponential Lower Bound

In this section we exhibit a class of planar acyclic digraphs which require exponential area in any planar straight-line upward drawing, for brevity, straight-line drawing.

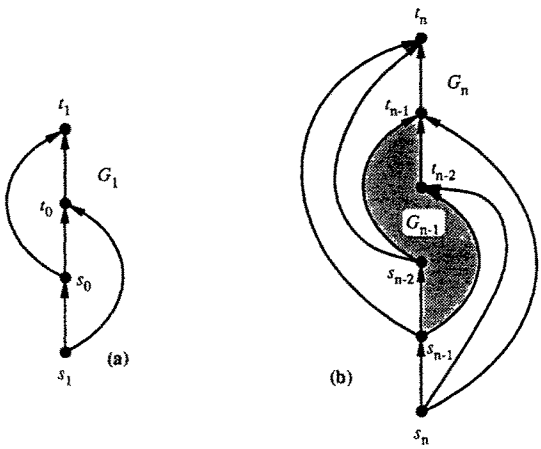

Fig. 2. A class of digraphs that require exponential area.

Let us define the following class of digraphs (see Fig. 2). Digraph G_1, shown in Fig. 2(a), consists of vertices s_0, s_1, t_0, and t_1 and edges (s_0, t_0), (s_1, s_0), (t_0, t_1), (s_1, t_0), and (s_0, t_1). For $n \geq 2$, digraph G_n is constructed from G_{n-1} by adding vertices s_n and t_n and edges (s_n, s_{n-1}), (t_{n-1}, t_n), (s_{n-2}, t_n), (s_n, t_{n-2}), (s_n, t_{n-1}), and (s_{n-1}, t_n), as shown in Fig. 2(b).

It is easy to verify that G_n is a planar $s_n t_n$-graph with $2n + 2$ vertices and $6n + 1$ edges. Also, G_n is triconnected for $n \geq 2$, and thus has a unique embedding. We show that the minimum area of a straight-line drawing of G_n is $\Omega(2^n)$ for every possible resolution rule.

Theorem 1. *Given any resolution rule, a planar straight-line upward drawing of digraph G_n (with $2n + 2$ vertices) has area $\Omega(2^n)$.*

Proof. Let A_n be the minimum area of a planar straight-line upward drawing of G_n. We use induction to prove that $A_n \geq 4 \cdot A_{n-2}$. Since $A_1 \geq c$, for some constant c depending on the resolution rule, this implies the claimed result.

Let Γ_n be a straight-line drawing of G_n with minimum area A_n. By removing from Γ_n vertices s_n and t_n and their incident edges, we obtain a straight-line drawing Γ_{n-1} of G_{n-1}. Also, by removing from Γ_{n-1} vertices s_{n-1} and t_{n-1} and their incident edges, we obtain a straight-line drawing Γ_{n-2} of G_{n-2}. Let σ and τ be horizontal lines through vertices s_{n-2} and t_{n-2}, respectively. Define θ_1 as the angle formed by edge (t_{n-3}, t_{n-2}) and the x-axis. Also, define θ_2 as the angle formed by edge (s_{n-2}, s_{n-3}) and the x-axis. We distinguish two cases:

Case 1: $\theta_1 \geq \theta_2$ (see Fig. 3). Let ρ_1 be the line extending edge (t_{n-3}, t_{n-2}) and let λ_1 be the line parallel to ρ_1 through vertex s_{n-2}. Also, let λ_2 be either the line extending edge (s_{n-2}, s_{n-3}) (Fig. 3(a)) or the line through vertices s_{n-2} and t_{n-2} (Fig. 3(b)), whichever forms the largest angle with the x-axis. Vertex s_{n-1} must lie in the region S_{n-1} below σ and to the right of ρ_1, since it is connected to vertices s_{n-2} and t_{n-2} from the right. Similarly, vertex t_{n-1} must lie in the region T_{n-1}

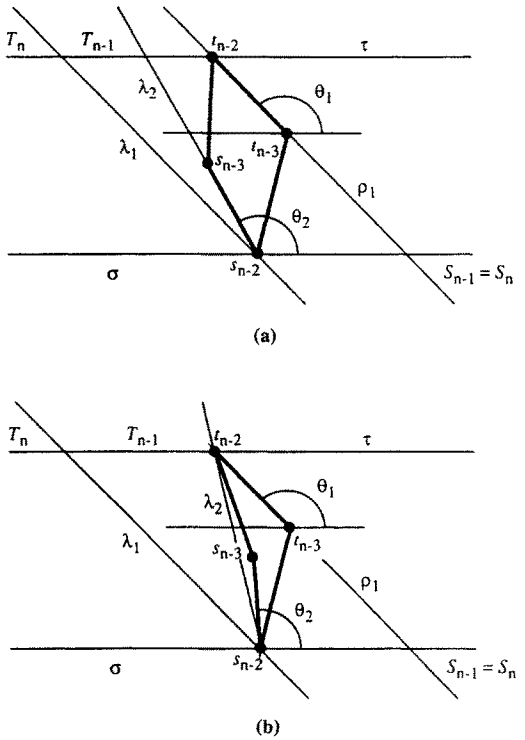

Fig. 3. Illustration of the proof of Theorem 1 for $\theta_1 \geq \theta_2$.

above τ and to the left of λ_2. With regard to vertices s_n and t_n we have that s_n must lie in the region $S_n = S_{n-1}$, since it is connected to t_{n-2} and t_{n-1} from the right, and t_n must lie in the region T_n above τ and to the left of λ_1, since it is connected to s_{n-1} from the left. (Actually, s_n and t_n must lie in proper subregions of S_{n-1} and T_{n-1}.)

Let P be the parallelogram delimited by lines σ, τ, λ_1, and ρ_1. Since s_{n-3} is vertically below t_{n-3}, the area of P is at least twice the area of Γ_{n-2}. Also, the area of Γ_{n-2} is greater than or equal to A_{n-2}, the minimum area required for drawing G_{n-2}. Hence,

$$Area(P) \geq 2 \cdot Area(\Gamma_{n-2}) \geq 2 \cdot A_{n-2}.$$

Now consider the triangle delimited by lines τ, ρ_1, and the line δ parallel to edge (s_{n-1}, t_n) through vertex s_{n-2} (see Fig. 4). Clearly, Γ_n must contain this triangle. Let Δ_1 be the triangle delimited by σ, ρ_1, and δ, and let Δ_2 be the triangle delimited by τ, λ_1, and δ. It follows that

$$A_n = Area(\Gamma_n) \geq Area(P) + Area(\Delta_1) + Area(\Delta_2).$$

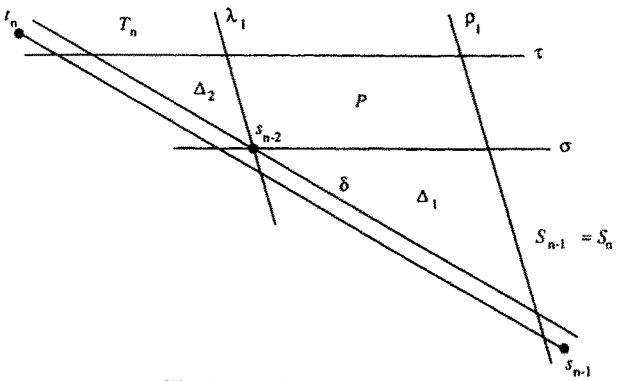

Fig. 4. Regions P, Δ_1, and Δ_2.

Since Δ_1 and Δ_2 are similar, the minimum of $Area(\Delta_1) + Area(\Delta_2)$ is equal to $Area(P)$. Hence we have

$$A_n \geq 2 \cdot Area(P) \geq 4 \cdot Area(\Gamma_{n-2}) = 4 \cdot A_{n-2}.$$

Case 2: $\theta_1 < \theta_2$ (see Fig. 5). The proof for this case is symmetric to the one for the previous case. In fact, notice that Fig. 5 is a 180° rotation of Fig. 3. □

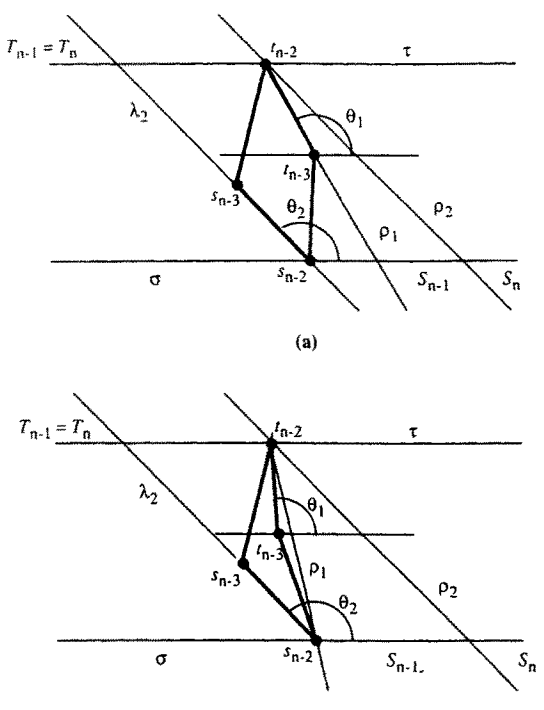

Fig. 5. Illustration of the proof of Theorem 1 for $\theta_1 \leq \theta_2$.

4. Algorithms for Drawing Planar Acyclic Digraphs

In this section we present a drawing algorithm for planar *st*-graphs that has several remarkable features: linear time complexity, small number of bends, small area, detection and display of symmetries, and geometric characterization of the transitive closure by means of the dominance relation between the points associated with the vertices. First we describe the algorithm for reduced digraphs, and then we extend it to the general case.

4.1. Reduced Digraphs

An edge (u, v) of a digraph is said to be *transitive* if there exists a directed path from u to v that does not contain the edge (u, v). An acyclic digraph is said to be *reduced* if it has no transitive edges. Notice that by removing all transitive edges from an acyclic digraph G we obtain a reduced digraph G^- that has the same transitive closure as the original digraph G. Reduced planar *st*-graphs are used in several applications, including VLSI layout compaction [17] and motion planning [15], [32], [41]. Also, they are important in the theory of partially ordered sets, because they represent the covering digraphs of planar lattices [21].

Let G be a reduced planar *st*-graph with vertex set V and edge set E. We recall that G is embedded in the plane with s and t on the external face. We denote with $u \rightarrow v$ a directed path from vertex u to vertex v in G (or the existence of such path). In this subsection we show how to construct a planar straight-line upward drawing of G.

A straight-line drawing of a digraph is a *dominance drawing* if, for any two vertices u and v, there is a directed path from u to v if and only if $x(u) \leq x(v)$ and $y(u) \leq y(v)$. Notice that these two conditions cannot be simultaneously satisfied with equality since distinct vertices must be placed at distinct points. Dominance drawings have the important feature of characterizing the transitive closure of the digraph by means of the geometric dominance relation among the vertices. A dominance drawing may have horizontal edges. In this case a counterclockwise rotation by any angle between $0°$ and $90°$ yields an upward drawing.

We present a lemma that characterizes the relation between dominance drawings and planarity; some terminology is needed. Given a dominance drawing Γ of G, we now consider the point $\Gamma(u) = (x(u), y(u))$ where vertex u is placed, and define the following four regions of the plane (see Fig. 6):

$$b(u) = \{(x, y): x \leq x(u) \text{ and } y \leq y(u)\},$$

$$t(u) = \{(x, y): x \geq x(u) \text{ and } y \geq y(u)\},$$

$$l(u) = \{(x, y): x < x(u) \text{ and } y > y(u)\},$$

$$r(u) = \{(x, y): x > x(u) \text{ and } y < y(u)\}.$$

Lemma 1. *Any dominance drawing Γ of a reduced planar st-graph G is planar.*

Proof. Suppose, for a contradiction, that there is a crossing between edges (u, v)

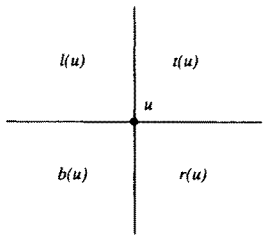

Fig. 6. Regions $b(u)$, $t(u)$, $l(u)$, and $r(u)$.

and (w, z) in Γ. Consider the edge (u, v): since G is reduced and Γ is a dominance drawing, no vertex p is placed in the rectangle defined by $\Gamma(u)$ and $\Gamma(v)$ (see Fig. 7). Otherwise, by the definition of dominance drawing, there would be paths $u \to p$ and $p \to v$, which implies that (u, v) is a transitive edge, thus contradicting the fact that G is reduced.

Without loss of generality, assume that $\Gamma(w, z)$ crosses $\Gamma(u, v)$ from left to right. First, $\Gamma(w)$ cannot be in $b(u)$, in fact in this case (w, z) would be transitive with respect to the path consisting of $w \to u$ and $u \to z$. Analogously, z cannot be in $t(v)$. Hence, the only possible case is that $w \in l(u) - l(v)$ and $z \in r(v) - r(u)$.

Consider paths $s \to u$ and $s \to w$, and let s' be the last (farthest from s) vertex common to such paths. Similarly, let t' be the first (farthest from t) vertex common to paths $v \to t$ and $z \to t$. By the above definitions and the dominance property, G has the following pairwise vertex-disjoint (except in the endpoints) paths (see Fig. 8):

$$s' \to w, \qquad s' \to u, \qquad v \to t', \qquad z \to t',$$
$$u \to v, \qquad w \to z, \qquad u \to z, \qquad w \to v.$$

Since s and t are on the external face, we can add to G the edge (s, t) while preserving planarity. It is easy to verify that the paths listed above plus the edge (s, t) form a graph that is homoeomorphic to $K_{3,3}$. This fact contradicts the planarity of G. □

Now we present the drawing algorithm. If not already given, a planar embedding of G can be constructed in $O(n)$ time using variations of well-known

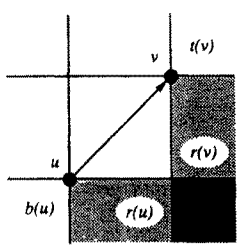

Fig. 7. Regions around edge (u, v) in the proof of Lemma 1.

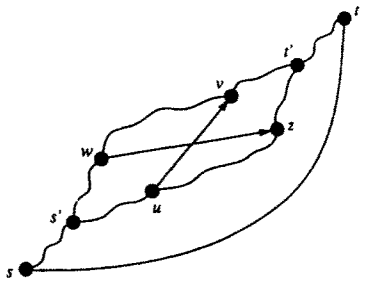

Fig. 8. A $K_{3,3}$ in the proof of Lemma 1.

planarity-testing algorithms [2], [14], [16], [23]. The algorithm consists of three phases: the first phase, *Preprocessing*, sets up a linked data structure; the second phase, *Preliminary Layout*, assigns to each vertex v a distinct X- and Y-coordinate in the range $[0, n-1]$; the third phase, *Compaction*, adjusts the position of the vertices to reduce the area of the drawing. The Preliminary Layout phase performs essentially two topological sortings of the vertices of G, which scan the successors of each vertex from left to right (e.g., clockwise) and from right to left (e.g., counterclockwise), respectively. It is a variation of the algorithm for constructing the "vector representation" of a planar st-graph described in [19]. The Compaction phase scans the vertices according to the order given by the preliminary X- and Y-coordinates.

Algorithm *Straight-Line-Draw*
Input: Reduced planar st-graph G.
Output: Dominance drawing Γ of G.
Preprocessing: Set up a linked data structure for G where each vertex v points to the list of its outgoing edges sorted according to their clockwise sequence around v. This list is doubly connected by means of pointers *next*(e) and *pred*(e), and is accessed by means of pointers *firstout*(v) and *lastout*(v) to its leftmost and rightmost edge, respectively. Also, v has pointers *firstin*(v) and *lastin*(v) to its leftmost and rightmost incoming edges, respectively. Finally each edge $e = (u, v)$ stores a pointer *head*(e) to its head-vertex v.
Preliminary Layout { Assign preliminary coordinates X and Y }

```
{ Assign preliminary coordinate X }
Set count := 0 and call LabelX(s):
procedure LabelX(v: vertex);
begin
  X(v) := count;
  count := count + 1;
  if v ≠ t then begin
    e := firstout(v);
    repeat
      w := head(e);
```

```
        if e = lastin(w)
          then LabelX(w);
        e := next(e);
      until e = nil
    end
end;
{ Assign preliminary coordinate Y }
Set count := 0 and call LabelY(s):
procedure LabelY(v: vertex);
begin
  Y(v) := count;
  count := count + 1;
  if v ≠ t then begin
    e := lastout(v);
    repeat
      w := head(e);
      if e = firstin(w)
        then LabelY(w);
      e := pred(e);
    until e = nil
  end
end;
```

Compaction { Assign final coordinates x and y }
Set-up two lists of vertices sorted by increasing X- and Y-coordinate by means of pointers $nextX(v)$ and $nextY(v)$.

```
{ Assign final coordinate x }
let u be the vertex with X(u) = 0;
x(u) := 0;
while nextX(u) ≠ nil do begin
  v := nextX(u);
  if Y(u) > Y(v) or (firstout(u) = lastout(u) and firstin(v) = lastin(v))
    then x(v) := x(u) + 1
    else x(v) := x(u);
  u := v;
end;
{ Assign final coordinate y }
let u be the vertex with Y(u) = 0;
y(u) := 0;
while nextY(u) ≠ nil do begin
  v := nextY(u);
  if X(u) > X(v) or (firstout(u) = lastout(u) and firstin(v) = lastin(v))
    then y(v) := y(u) + 1
    else y(v) := y(u);
  u := v;
end;
```

Let u and v be a pair of vertices with consecutive (preliminary) X-coordinates. In general, the (final) x-coordinate is not incremented if (u, v) is an edge, and is incremented otherwise. However, in the special case when (u, v) is the only outgoing edge of u and the only incoming edge of v, the x-coordinate is incremented. This is done to prevent the possibility that u and v be assigned the same pair of coordinates. Similar considerations can be made for the y-coordinates.

Algorithm *Straight-Line-Draw* is simple to implement and appears to be eminently practical. A run of Algorithm *Straight-Line-Draw* is illustrated in Fig. 9. The preliminary drawing (X- and Y-coordinates) is shown in Fig. 9(a) (arrows are omitted because they are implied by the upward requirement). The final drawing (x- and y-coordinates) is shown in Fig. 9(b). Perhaps the best aesthetic result is obtained by a 45° rotation, as shown in Fig. 9(c). Given a vertex u of G we define $B(u)$ (resp. $T(u)$) as the set of vertices distinct from u that can reach (resp. can be reached from) u by a directed path. Also, we define $L(u)$ (resp. $R(u)$) as the set of vertices that are on the left (resp. right) of every path from s to t through u (see Fig. 10). Note that $\{u\}$, $B(u)$, $T(u)$, $L(u)$, and $R(u)$ form a partition of the vertices of G.

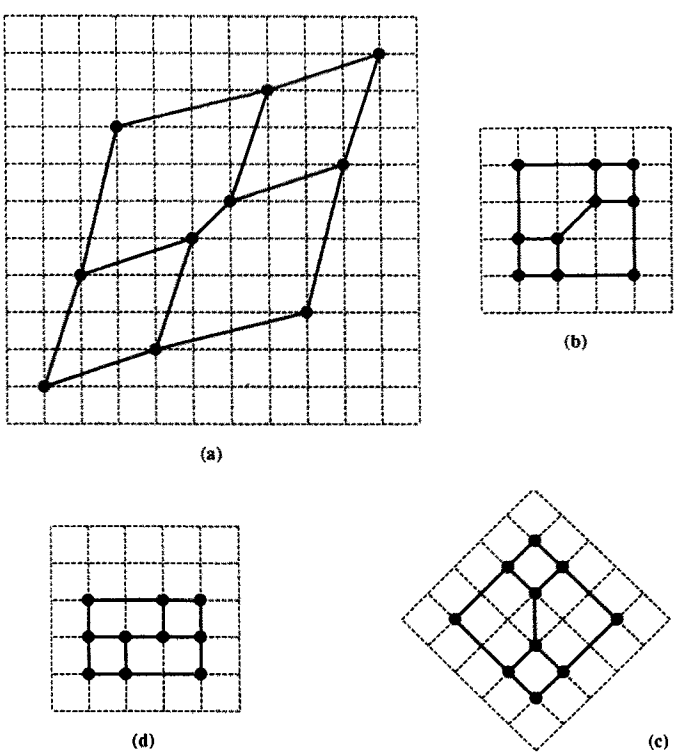

Fig. 9. A run of Algorithm *Straight-Line-Draw*: (a) preliminary drawing; (b) final drawing; (c) final drawing rotated by 45°; (d) minimum area drawing.

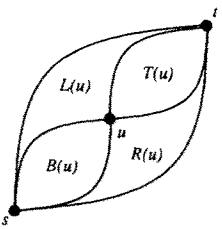

Fig. 10. Vertex sets $B(u)$, $T(u)$, $L(u)$, and $R(u)$.

Lemma 2. *The X- and Y-coordinates computed in the Preliminary Layout phase of Algorithm* Straight-Line-Draw *have the following properties:*

1. $X(u) < X(v)$ *if and only if* $u \in B(v) \cup L(v)$.
2. $Y(u) < Y(v)$ *if and only if* $u \in B(v) \cup R(v)$.

Proof. We give the proof of property 1. A similar argument holds for property 2. For the "if" part, observe that the recursive calls of procedure LabelX define a directed spanning tree T of G rooted at s and containing the rightmost incoming edge of each vertex. Suppose, for a contradiction, that $X(u) > X(v)$. Since vertex u is visited after vertex v by LabelX, vertex u is either a descendant of v in T, or it is on a path to the right of the path π from s to v in T. In the first case we have $v \to u$, which contradicts the hypothesis that $u \in B(v) \cup L(v)$. In the second case vertex u cannot be in $B(v)$, because no directed path in G can enter vertex v to the right of the path π; also, vertex u cannot be in $L(v)$, since it is to the right of the path of G from s to t obtained by extending π with leftmost outgoing edges. The "only-if" part follows from the fact that $u \in B(v) \cup L(v)$ if and only if $v \in T(u) \cup R(u)$. □

Theorem 2. *The drawing* Π *of G described by the X- and Y-coordinates computed in the Preliminary Layout phase of Algorithm* Straight-Line-Draw *is a dominance drawing.*

Proof. By Lemma 2, we have $u \in B(v) \Leftrightarrow X(u) \leq X(v)$ **and** $Y(u) \leq Y(v)$. □

Lemma 3. *Let u and v be a pair of vertices of G such that* $X(v) = X(u) + 1$. *Then* $Y(u) < Y(v)$ *if and only if G has an edge from u to v.*

Proof. The "if" part is trivial. For the "only-if" part, suppose that $Y(u) < Y(v)$. Since $X(u) < X(v)$, we have $v \in T(u)$. If (u, v) is not an edge, then the path $(u \to v)$ has a vertex w distinct from u and from v. Hence, $X(u) < X(w) < X(v)$, thus contradicting the hypothesis that $X(v) = X(u) + 1$. □

Theorem 3. *Let G be a reduced planar st-graph with n vertices. Algorithm* Straight-Line-Draw *takes* $O(n)$ *time and constructs a planar dominance drawing* Γ *of G with vertices placed at grid points and* $O(n^2)$ *area.*

Proof. Let Π be the drawing given by the X- and Y-coordinates. By Theorem 2, Π is a dominance drawing. To prove that the drawing Γ given by the x- and y-coordinates is also a dominance drawing we show that:

1. $u \in B(v) \Rightarrow x(u) \leq x(v)$ **and** $y(u) \leq y(v)$.
2. $x(u) < x(v) \Rightarrow u \in B(v) \cup L(v)$.
3. $y(u) < y(v) \Rightarrow u \in B(v) \cup R(v)$.
4. $x(u) = x(v) \Rightarrow u \in B(v) \cup T(v)$.
5. $y(u) = y(v) \Rightarrow u \in B(v) \cup T(v)$.
6. No two vertices are drawn at the same point (x, y).

It is immediate to verify that, for any two vertices u and v, we have

$$\begin{aligned} X(u) < X(v) &\Rightarrow x(u) \leq x(v), \\ Y(u) < Y(v) &\Rightarrow y(u) \leq y(v), \\ x(u) < x(v) &\Rightarrow X(u) < X(v), \\ y(u) < y(v) &\Rightarrow Y(u) < Y(v). \end{aligned}$$

Hence, properties 1–3 follow by Lemma 2. Assume that $X(u) < X(v)$ and $x(u) = x(v)$. Let $u = w_1, w_2, \ldots, w_k = v$ be the sequence of vertices with X-coordinate in the range $[X(u), X(v)]$. Since the x-coordinate is not incremented on these vertices, we must have $Y(w_1) < Y(w_2) < \cdots < Y(w_k)$, and hence $Y(u) < Y(v)$. Since Π is a dominance drawing, we have that $u \in B(v)$. Thus property 4 is verified, and a similar argument proves property 5.

Regarding property 6, assume, for a contradiction, that it does not hold and there are vertices u and v with $x(u) = x(v)$ and $y(u) = y(v)$. By properties 1–4, we may assume that $X(u) < X(v)$ and $Y(u) < Y(v)$. By Lemma 3, all the vertices w such that $X(u) \leq X(w) \leq X(v)$ form a path from u to v, and all the vertices with Y-coordinate between $Y(u)$ and $Y(v)$ form exactly the same path. Let z be the vertex such that $X(z) = X(v) - 1$ and $Y(z) = Y(v) - 1$. We have that $x(z) = x(v)$ and $y(z) = y(v)$. Since both procedures LabelX and LabelY visit v immediately after z, edge (z, v) must be the only outgoing edge of z and the only incoming edge of v. However, this causes the Compaction phase to increment both x and y at vertex v, thus contradicting the previous conclusion that $x(z) = x(v)$ and $y(z) = y(v)$.

The area of the drawing Γ is given by $x(t) \cdot y(t)$. Consider the assignment of the x-coordinates. At the end of the Preliminary Layout phase we have $X(t) = Y(t) = n - 1$. The Compaction step scans the X-list from s to t and, for each pair of consecutive vertices u and v, either $x(v) = x(u)$ or $x(v) = x(u) + 1$. Hence, since $x(s) = 0$, $x(t) \leq n - 1$. Similar arguments show that $y(t) \leq n - 1$.

Concerning the time complexity, procedures LabelX and LabelY traverse each edge twice. At the beginning of the Compaction phase the two lists can be constructed using a bucket sort. The remaining while-loops take linear time to scan the lists and perform a constant-time test for each vertex. □

The correctness of the algorithm can also be proved by exploiting the fact that the partial order underlying a planar lattice has *dimension* two, i.e., it can be

generated by the intersection of two linear extensions [19], [21], [22]. Notice that a quadratic bound on the area is asymptotically optimal even if bends are allowed [47]. A tighter bound on the area is presented in Section 4.3.

4.2. *Display of Symmetries*

Besides producing a dominance drawing, Algorithm *Straight-Line-Draw* has the important feature of displaying the symmetries and isomorphic components of the digraph. Before describing these features we introduce some definitions on symmetries of planar *st*-graphs.

A digraph G is *weakly connected* if its underlying undirected graph is connected. Let G be a planar *st*-graph. An *open component* of G is a maximal weakly connected subgraph G' of the digraph obtained from G by removing a separation pair $\{p, q\}$, such that G' does not contain s or t. A *closed component* of G is an induced subgraph G' of G such that (see Fig. 11(a)):

1. G' is a planar pq-graph.
2. G' contains every vertex of G that is on some path from p to q.
3. G' contains every outgoing edge of p, every incoming edge of q, and every incident edge of the remaining vertices of G'.

A *component* of G is either a closed or an open component. Notice that G is a trivial closed component of itself.

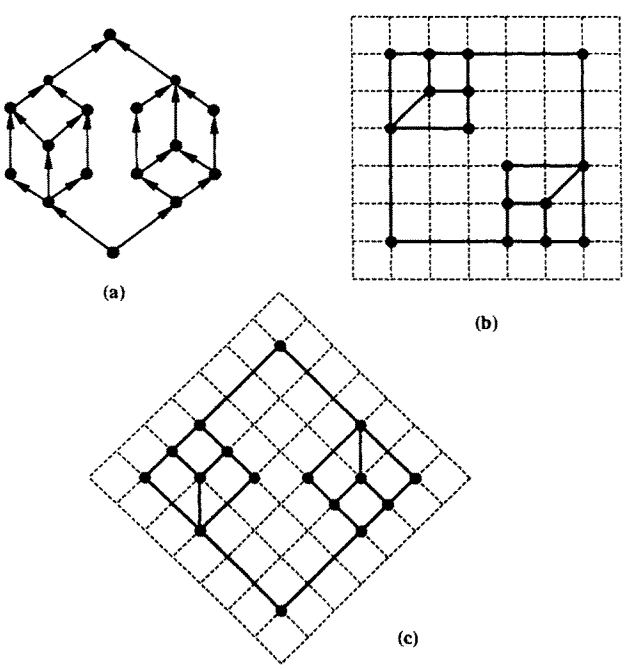

Fig. 11. (a) A planar *st*-graph G with two rotationally isomorphic components. (b) Drawing of G constructed by Algorithm *Straight-Line-Draw*. (c) Rotated drawing.

The digraph obtained from a closed component by removing its source and sink is not necessarily an open component, but, in general, the union of several open components. Also, the digraph obtained from an open component by adding the separation pair is not necessarily a closed component, since properties 2 and 3 above might not be verified. The concept of an open component generalizes the one of a subtree of a rooted tree, as follows. Let T be a tree rooted at vertex s. We construct a planar st-graph G_T by connecting all the leaves of T to a new vertex t. It is simple to verify that the subtree of T rooted at a vertex $v \neq s$ is an open component of G_T.

Let C_1 and C_2 be two components of a planar st-graph G that are isomorphic if we ignore the directions of the edges. C_1 and C_2 are said to be *simply isomorphic* if the isomorphism preserves the directions of the edges and the clockwise boundaries of the faces. C_1 and C_2 are said to be *axially isomorphic* if the isomorphism preserves the directions of the edges and inverts the clockwise boundaries of the faces. C_1 and C_2 are said to be *rotationally isomorphic* if the isomorphism inverts the directions of the edges and preserves the clockwise boundaries of the faces. A component is said to be *axially (rotationally) symmetric* if it is axially (rotationally) isomorphic to itself. For example, the digraph of Fig. 11(a) has two rotationally isomorphic components, and each such component is axially symmetric. Figures 11(b) and (c) show the drawing produced by Algorithm *Straight-Line-Draw* for the digraph of Fig. 11(a).

Let E_L be the set of edges (u, v) such that (u, v) is the rightmost incoming edge of v and the leftmost outgoing edge of u. Let E_R be the set of edges (u, v) such that (u, v) is the leftmost incoming edge of v and the rightmost outgoing edge of u. Also, we define E_H as the set of edges (u, v) such that (u, v) is the only outgoing edge of u and the only incoming edge of v. Observe that $E_H = E_L \cap E_R$. We write $m_L = |E_L|$, $m_R = |E_R|$, and $m_H = |E_H|$. In the example of Fig. 9 the set E_H contains exactly one edge.

Lemma 4. *Let $(u, v) \in E$. Then u and v appear consecutively in the X-list if and only if $(u, v) \in E_L$. Also, u and v appear consecutively in the Y-list if and only if $(u, v) \in E_R$.*

By Lemmas 3 and 4 the tests for incrementing the x and y-coordinates in the Compaction step can be rewritten as follows:

```
if (u, v) ∈ E_L − E_H
   then x(v) := x(u)
   else x(v) := x(u) + 1;
if (u, v) ∈ E_R − E_H
   then y(v) := y(u)
   else y(v) := y(u) + 1;
```

Theorem 4. *Let G be a reduced planar st-graph and let Γ be the corresponding straight-line drawing constructed by Algorithm* Straight-Line-Draw. *We have:*

1. *Simply isomorphic components of G have drawings in Γ that are congruent up to a translation.*

2. *Axially isomorphic components of G have drawings in* Γ *that are congruent up to a translation and reflection.*
3. *Rotationally isomorphic components of G have drawings in* Γ *that are congruent up to a translation and a* 180° *rotation.*
4. *The drawing of an axially symmetric component of G is symmetric with respect to the straight line that passes through its source and sink.*
5. *The drawing of a rotationally symmetric component of G is symmetric with respect to a* 180° *rotation around its centroid.*

Proof. In the Preprocessing phase the vertices of a component are visited consecutively by procedures LabelX and LabelY. Hence, the layout of a component is independent from the rest of the digraph. This proves property 1. With regard to properties 2 and 4, reversing the orientation of the faces exchanges the set $L(u)$ with $R(u)$ for every vertex u. By Lemma 2, this corresponds to exchanging the X-coordinate with the Y-coordinate, and similarly for the final x- and y-coordinates. This yields drawings that are congruent up to a translation and a reflection with respect to a 45°-slope line. Now we consider properties 3 and 5. Reversing the direction of the edges exchanges $B(u)$ with $T(u)$ and $L(u)$ with $R(u)$ for every vertex u. Hence, by Lemma 2, the X-lists of two rotationally isomorphic components are one the reverse of the other, and similarly for the Y-lists. This implies that properties 3 and 5 hold for the preliminary layout. The sets E_L, E_H, and E_R stay the same after reversing the direction of the edges. Thus, the final x- and y-coordinates are incremented for the same pairs of vertices and properties 3 and 5 hold for the final layout. □

4.3. *Minimum-Area Drawings*

As shown in Theorem 3, Algorithm *Straight-Line-Draw* produces drawings with $O(n^2)$ area. Here we give a tighter upper bound on the area and present a modification of the algorithm that constructs a minimum-area drawing among all dominance drawings of G. We can express $x(t)$ and $y(t)$ in terms of n, m_L, m_R, and m_H as follows:

Lemma 5. $x(t) = n - 1 - (m_L - m_H)$ *and* $y(t) = n - 1 - (m_R - m_H)$.

Proof. The number of times that the x-coordinate (resp. y coordinate) is *not* incremented is equal to $m_L - m_H$ (resp. $m_R - m_H$). □

Recalling that the area of the drawing constructed by Algorithm *Straight-Line-Draw* is given by $x(t) \cdot y(t)$, we obtain the following tight bound.

Theorem 5. *Algorithm* Straight-Line-Draw *produces drawings with area*

$$(n - 1 - (m_L - m_H)) \times (n - 1 - (m_R - m_H)).$$

Suppose that $E_H = \varnothing$. In this case we can prove that the area of the drawing is optimal. Next we show how to modify the algorithm to obtain a minimum-area drawing in the case when $E_H \neq \varnothing$.

Theorem 6. *Given a reduced planar st-graph G such that $E_H = \emptyset$, Algorithm* Straight-Line-Draw *produces a dominance drawing of G that has minimum area among all dominance drawings that place vertices at grid points and preserve the embedding of G.*

Proof. First we observe that any dominance drawing that preserves the embedding of G must place the vertices of $L(v)$ in $l(v)$ and the vertices of $R(v)$ in $r(v)$. Let v_i be the vertex that is assigned X-coordinate i by the Preliminary Layout phase of Algorithm *Straight-Line-Draw*. By Lemma 2, in any drawing of G we have $x(v_i) \leq x(v_{i+1})$. Now consider the $n - m_L$ pairs of vertices $\{v_i, v_{i+1}\}$ such that $Y(v_i) > Y(v_{i+1})$. By Lemma 2, $v_{i+1} \in R(v_i)$ so that we must have $x(v_{i+1}) > x(v_i)$. We conclude that $x(v_{n-1}) - x(v_0) \geq n - m_L - 1$. A similar argument shows that $y(v_{n-1}) - y(v_0) \geq n - m_R - 1$. Hence, by Theorem 5, the drawing constructed by Algorithm *Straight-Line-Draw* has optimal area. □

Theorem 7. *Let G be a reduced planar st-graph with n vertices. A minimum-area dominance drawing of G that places vertices at grid points and preserves the embedding of G can be constructed in $O(n)$ time.*

Proof. To take into account the set E_H, we use the following variation of Algorithm *Straight-Line-Draw*. In the Preprocessing phase we compute m_L and m_R. In the Compaction phase we replace the first "if" test with

if $Y(u) > Y(v)$ **or** (*firstout*(u) = *lastout*(u) **and** *firstin*(v) = *lastin*(v) **and** $m_L \leq m_R$)

and the second "if" test with

if $X(u) > X(v)$ **or** (*firstout*(u) = *lastout*(u) **and** *firstin*(v) = *lastin*(v) **and** $m_L > m_R$).

This yields a drawing with area

$$A = (n - 1 - \min(m_L, m_R) + m_H) \times (n - 1 - \max(m_L, m_R)).$$

Clearly, for any dominance drawing of G, we must have an increment of the x or y coordinate in correspondence of every edge of E_H. If m_x and m_y are respectively the increments of x and y, with $m_x + m_y = m_H$, the area is at least

$$(n - 1 - m_L + m_x) \times (n - 1 - m_R + m_y).$$

It is easy to see that the minimum of the above quantity is equal to A, and is achieved by setting

$$m_x = \begin{cases} m_H & \text{if } m_L \leq m_R, \\ 0 & \text{if } m_L > m_R. \end{cases}$$

□

Note that minimum-area drawings may not have the symmetry properties of Theorem 4.

4.4. *General Planar st-Graphs*

Algorithm *Straight-Line-Draw* can be extended to general planar *st*-graphs by inserting a new dummy vertex on every transitive edge.

Algorithm *Polyline-Draw*
Input: Planar *st*-graph G.
Output: Planar polyline upward drawing Γ of G.

1. If G is not reduced, replace each transitive edge (u, v) with a chain of length two consisting of a new vertex, x, and two new edges, (u, x) and (x, v). Let G' be the resulting reduced planar *st*-graph.
2. Construct a straight-line drawing Γ' of G' using Algorithm *Straight-Line-Draw*.
3. A polyline drawing Γ of the original digraph G is finally obtained by considering the dummy vertices of Γ' as bends of Γ.

Since the number of transitive edges is at most $2n - 5$, we have

Theorem 8. *Let G be a planar st-graph with n vertices. Algorithm* Polyline-Draw *has $O(n)$ time complexity and constructs a planar polyline upward drawing Γ of G with vertices placed at grid points, $O(n^2)$ area, and at most $2n - 5$ bends. Also, the drawing has all the symmetry properties* 1–5 *of Theorem* 4.

5. Open Problems

We conclude the paper with the following open problems:

1. Find a polynomial-time algorithm for testing whether a digraph G admits a planar upward drawing (i.e., whether G is a subgraph of a planar *st*-graph [6]), or show that the problem is NP-complete. Polynomial-time algorithms exist for special classes of digraphs, namely bipartite [5], triconnected [1], and single-source [18].
2. Give upper bounds on the area of planar straight-line upward drawings with vertices placed at grid points.
3. Study the tradeoff between area and number of bends in polyline planar upward drawings.
4. Is there an $O(n)$-time algorithm for constructing a straight-line planar upward drawing of an n-vertex planar *st*-graph?

References

1. P. Bertolazzi and G. Di Battista, On upward drawing testing of triconnected digraphs, *Proceedings of the ACM Symposium on Computational Geometry*, 1991, pp. 272–280.
2. N. Chiba, T. Nishizeki, S. Abe, and T. Ozawa, A linear algorithm for embedding planar graphs using PQ-trees, *Journal of Computer and System Sciences* **30** (1985), 54–76.

3. N. Chiba, K. Onoguchi, and T. Nishizeki, Drawing planar graphs nicely, *Acta Informatica* **22** (1985), 187–201.
4. M. Chrobak, A linear-time algorithm for drawing a planar graph on a grid, Manuscript, University of California, Riverside, 1988.
5. G. Di Battista, W.-P. Liu, and I. Rival, Bipartite graphs, upward drawings, and planarity, *Information Processing Letters* **36** (1990), 317–322.
6. G. Di Battista and R. Tamassia, Algorithms for plane representations of acyclic digraphs, *Theoretical Computer Science* **61** (1988), 175–198.
7. D. Dolev, F. T. Leighton, and H. Trickey, Planar embedding of planar graphs, in *Advances in Computing Research*, vol. 2, F. P. Preparata, ed., JAI Press, Greenwich, CT, 1984, pp. 147–161.
8. P. Eades, A heuristic for graph drawing, *Congressus Numerantium* **42** (1984), 149–160.
9. P. Eades and R. Tamassia, Algorithms for automatic graph drawing: An annotated bibliography, Technical Report CS-89-09, Dept. of Computer Science, Brown University, 1989.
10. P. Eades and R. Tamassia, Algorithms for automatic graph drawing: An annotated bibliography, *Networks* (1992), to appear.
11. I. Fary, On straight lines representation of planar graphs, *Acta Scientiarum Mathematicarum (Szeged)* **11** (1948), 229–233.
12. H. de Fraysseix, J. Pach, and R. Pollack, Small sets supporting Fary embeddings of planar graphs, *Proceedings of the 20th ACM Symposium on Theory of Computing*, 1988, pp. 426–433.
13. H. de Fraysseix, J. Pach, and R. Pollack, How to draw a planar graph on a grid, *Combinatorica* **10** (1990), 41–51.
14. H. de Fraysseix and P. Rosenstiehl, A depth-first-search characterization of planarity, *Annals of Discrete Mathematics* **13** (1982), 75–80.
15. L. J. Guibas and F. F. Yao, On translating a set of rectangles, in *Advances in Computing Research*, vol. 1, F. P. Preparata, ed., JAI Press, Greenwich, CT, 1983, pp. 61–77.
16. J. Hopcroft and R. E. Tarjan, Efficient planarity testing, *Journal of the Association for Computing Machinery* **21** (1974), 549–568.
17. M. Y. Hsueh and D. O. Pederson, Computer-aided layout of LSI circuit building-blocks, *Proceedings of the IEEE International Symposium on Circuits and Systems*, 1979, pp. 474–477.
18. M. D. Hutton and A. Lubiw, Upward planar drawing of single source acyclic digraphs, *Proceedings of the ACM–SIAM Symposium on Discrete Algorithms*, 1991, pp. 203–211.
19. T. Kameda, On the vector representation of the reachability in planar directed graphs, *Information Processing Letters* **3** (1975), 75–77.
20. D. Kelly, Fundamentals of planar ordered sets, *Discrete Mathematics* **63** (1987), 197–216.
21. D. Kelly and I. Rival, Planar lattices, *Canadian Journal of Mathematics* **27** (1975), 636–665.
22. D. Kelly and W. T. Trotter, Dimension theory for ordered sets, in *Ordered Sets*, I. Rival, ed., Reidel, Dordrecht, 1982, pp. 171–211.
23. A. Lempel, S. Even, and I. Cederbaum, An algorithm for planarity testing of graphs, *Theory of Graphs* (Proc. International Symposium), Gordon and Breach, New York, 1967, pp. 215–232.
24. R. Lipton, S. North, and J. Sandberg, A method for drawing graphs, *Proceedings of the ACM Symposium on Computational Geometry*, 1985, pp. 153–160.
25. J. Manning, Computational complexity of geometric symmetry detection in graphs, Technical Report CSC-90-1, Dept. of Computer Science, University of Missouri, Rolla, 1990.
26. J. Manning and M. J. Atallah, Fast detection and display of symmetry in outerplanar graphs, Technical Report CSD-TR-606, Dept. of Computer Sciences, Purdue University, West Lafayette, 1986.
27. J. Manning and M. J. Atallah, Fast detection and display of symmetry in trees, *Congressus Numerantium* **64** (1988), 159–169.
28. C. Platt, Planar lattices and planar graphs, *Journal of Combinatorial Theory, Series B* **21** (1976), 30–39.
29. R. Read, New methods for drawing a planar graph given the cyclic order of the edges at Each Vertex, *Congressus Numerantium* **56** (1987), 31–44.
30. E. Reingold and J. Tilford, Tidier drawing of trees, *IEEE Transactions on Software Engineering* **7** (1981), 223–228.

31. I. Rival, Graphical data structures for ordered sets, in *Algorithms and Order*, I. Rival, ed., Kluwer, Boston, 1989, pp. 3–31.
32. I. Rival and J. Urrutia, Representing orders by translating convex figures in the plane, *Order* **4** (1988), 319–339.
33. P. Rosenstiehl and R. E. Tarjan, Rectilinear planar layouts of planar graphs and bipolar orientations, *Discrete & Computational Geometry* **1** (1986), 343–353.
34. W. Schnyder, Embedding planar graphs on the grid, *Proceedings of the ACM–SIAM Symposium on Discrete Algorithms*, 1990, pp. 138–148.
35. S. K. Stein, Convex maps, *Proceedings of the American Mathematical Society* **2** (1951), 464–466.
36. E. Steinitz and H. Rademacher, *Vorlesung uber die Theorie der Polyeder*, Springer-Verlag, Berlin, 1934.
37. J. A. Storer, On minimal node-cost planar embeddings, *Networks* **14** (1984), 181–212.
38. K. J. Supowit and E. M. Reingold, The complexity of drawing trees nicely, *Acta Informatica* **18** (1983), 377–392.
39. R. Tamassia, On embedding a graph in the grid with the minimum number of bends, *SIAM Journal on Computing* **16** (1987), 421–444.
40. R. Tamassia, G. Di Battista, and C. Batini, Automatic graph drawing and readability of diagrams, *IEEE Transactions on Systems, Man and Cybernetics* **18** (1988), 61–79.
41. R. Tamassia and F. P. Preparata, Dynamic maintenance of planar digraphs, with applications, *Algorithmica* **5** (1990), 509–527.
42. R. Tamassia and I. G. Tollis, A unified approach to visibility representations of planar graphs, *Discrete & Computational Geometry* **1** (1986), 321–341.
43. R. Tamassia and I. G. Tollis, Planar grid embedding in linear time, *IEEE Transactions on Circuits and Systems* **36** (1989), 1230–1234.
44. C. Thomassen, Planar acyclic oriented graphs, *Order* **5** (1989), 349–361.
45. W. T. Tutte, How to draw a graph, *Proceedings of the London Mathematical Society* **3** (1963), 743–768.
46. K. Wagner, Bemerkungen zum Vierfarbenproblem, *Jahresbericht der Deutschen Mathematiker-Vereinigung* **46** (1936), 26–32.
47. D. Woods, Drawing planar graphs, Ph.D. dissertation (Technical Report STAN-CS-82-943), Computer Science Dept., Stanford University, 1982.

Received April 14, 1990, and in revised form May 24, 1991.