# 44 GRAPH DRAWING

Roberto Tamassia

## INTRODUCTION

Graph drawing addresses the problem of constructing geometric representations of graphs, and has important applications to key computer technologies such as software engineering, database systems, visual interfaces, and computer-aided-design. Research on graph drawing has been conducted within several diverse areas, including discrete mathematics (topological graph theory, geometric graph theory, order theory), algorithmics (graph algorithms, data structures, computational geometry, VLSI), and human-computer interaction (visual languages, graphical user interfaces, software visualization). This chapter overviews aspects of graph drawing that are especially relevant to computational geometry. Basic definitions on drawings and their properties are given in Section 44.1. Bounds on geometric and topological properties of drawings (e.g., area and crossings) are presented in Section 44.2. Section 44.3 deals with the time complexity of fundamental graph drawing problems. An example of a drawing algorithm is given in Section 44.4. General techniques for drawing graphs are surveyed in Section 44.5. Section 44.6 covers selected topics that have recently attracted considerable research interest.

# 44.1 DRAWINGS AND THEIR PROPERTIES

## TYPES OF GRAPHS

First, we define some terminology on graphs pertinent to graph drawing. Throughout let $n$ and $m$ be the number of graph vertices and edges respectively, and $d$ the maximum vertex degree (i.e., number of incident edges).

## GLOSSARY

***Degree-k graph:*** Graph with maximum degree $d \leq k$.

***Digraph:*** Directed graph, i.e., graph with directed edges (drawn as arrows).

***Acyclic digraph:*** Without directed cycles.

***Transitive edge:*** Edge $(u, v)$ of a digraph is transitive if there is a directed path from $u$ to $v$ not containing edge $(u, v)$.

***Reduced digraph:*** Without transitive edges.

***Source:*** Vertex of a digraph without incoming edges.

***Sink:*** Vertex of a digraph without outgoing edges.

***st-digraph:*** Acyclic digraph with exactly one source and one sink, which joined by an edge (also called ***bipolar digraph***).

***Connected graph:*** Any two vertices are joined by a path.

***Biconnected graph:*** Any two vertices are joined by two vertex-disjoint paths.

***Triconnected graph:*** Any two vertices are joined by three vertex-disjoint paths.

***Tree:*** Connected graph without cycles.

***Rooted tree:*** Directed tree with a distinguished vertex, the ***root***, such that each vertex lies on a directed path to the root.

***Binary tree:*** Rooted tree where each vertex has at most two incoming edges.

***Layered (di)graph:*** The vertices are partitioned into sets, called layers. A rooted tree can be viewed as a layered digraph where the layers are sets of vertices at the same distance from the root.

***k-layered (di)graph:*** Layered (di)graph with $k$ layers.

## TYPES OF DRAWINGS

In a drawing of a graph, vertices are represented by points (or by geometric figures such as circles or rectangles) and edges are represented by curves such that any two edges intersect at most in a finite number of points. Except for Section 44.6, which covers three-dimensional drawings, we consider drawings in the plane.

## GLOSSARY

***Polyline drawing:*** Each edge is a polygonal chain (Fig. 44.1.1(a)).

***Straight-line drawing:*** Each edge is a straight-line segment (Fig. 44.1.1(b)).

***Orthogonal drawing:*** Each edge is a chain of horizontal and vertical segments (Fig. 44.1.1(c)).

***Bend:*** In a polyline drawing, point where two segments part of the same edge meet (Fig. 44.1.1(a)).

***Crossing:*** Point where two edges intersect (Fig. 44.1.1(b)).

***Grid drawing:*** Polyline drawing such that vertices, crossings and bends have integer coordinates.

***Planar drawing:*** No two edges cross (see Fig. 44.1.1(d)).

***Planar (di)graph:*** Admits a planar drawing.

***Embedded (di)graph:*** Planar (di)graph with a prespecified topological embedding (i.e., set of faces), which must be preserved in the drawing.

***Upward drawing:*** Drawing of a digraph where each edge is monotonically nondecreasing in the vertical direction (see Fig. 44.1.1(d)).

***Upward planar digraph:*** Admits an upward planar drawing.

***Layered drawing:*** Drawing of a layered graph such that vertices in the same layer lie on the same horizontal line (also called ***hierarchical drawing***).

***Face:*** A region of the plane defined by a planar drawing, where the unbounded region is called the ***external face***.

***Convex drawing:*** Planar straight-line drawing such that the boundary of each face is a convex polygon.

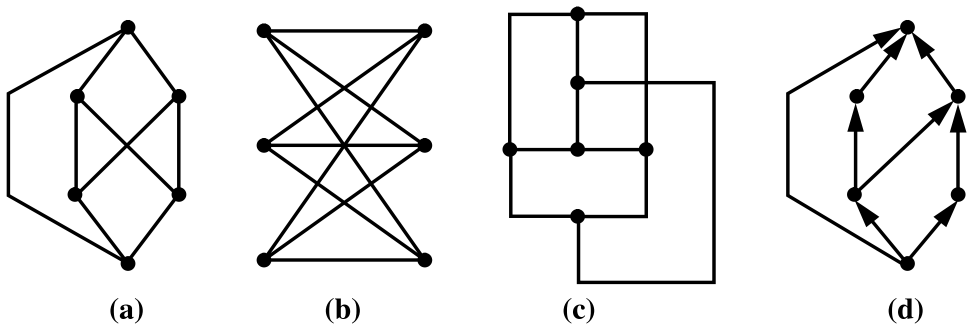
***Visibility drawing:*** Drawing of a graph based on a geometric visibility relation. E.g., the vertices might be drawn as horizontal segments, and the edges associated with vertically visible segments.

***Proximity drawing:*** Drawing of a graph based on a geometric proximity relation. E.g., a tree is drawn as the Euclidean minimum spanning tree of a set of points.

***Dominance drawing:*** Upward drawing of an acyclic digraph such that there exists a directed path from vertex $u$ to vertex $v$ if and only if $x(u) \leq x(v)$ and $y(u) \leq y(v)$, where $x(\cdot)$ and $y(\cdot)$ denote the coordinates of a vertex.

***hv-drawing:*** Upward orthogonal straight-line drawing of a binary tree such that the drawings of the subtrees of each node are separated by a horizontal or vertical line.

FIGURE 44.1.1
*Types of drawings: (a) polyline drawing of $K_{3,3}$; (b) straight-line drawing of $K_{3,3}$; (c) orthogonal drawing of $K_{3,3}$; (d) planar upward drawing of an acyclic digraph.*




Straight-line and orthogonal drawings are special cases of polyline drawings. Polyline drawings provide great flexibility since they can approximate drawings with curved edges. However, edges with more than two or three bends may be difficult to "follow" for the eye. Also, a system that supports editing of polyline drawings is more complicated than one limited to straight-line drawings. Hence, depending on the application, polyline or straight-line drawings may be preferred. If vertices are represented by points, orthogonal drawings exist only for graphs of maximum vertex degree 4.

## PROPERTIES OF DRAWINGS

## GLOSSARY

***Crossings ($\chi$):*** Total number of crossings in a drawing.

***Area:*** Area of the convex hull of the drawing.

***Total edge length:*** Sum of the lengths of the edges.

***Maximum edge length:*** Maximum length of an edge.

***Total number of bends:*** Total number of bends on the edges of a polyline drawing.

***Maximum number of bends:*** Maximum number of bends on an edge of a polyline drawing.

***Angular resolution ($\rho$):*** Smallest angle formed by two edges, or segments of edges, incident on the same vertex or bend, in a polyline drawing.

***Aspect-ratio:*** Ratio of the longest to the shortest side of the smallest rectangle with horizontal and vertical sides covering the drawing.

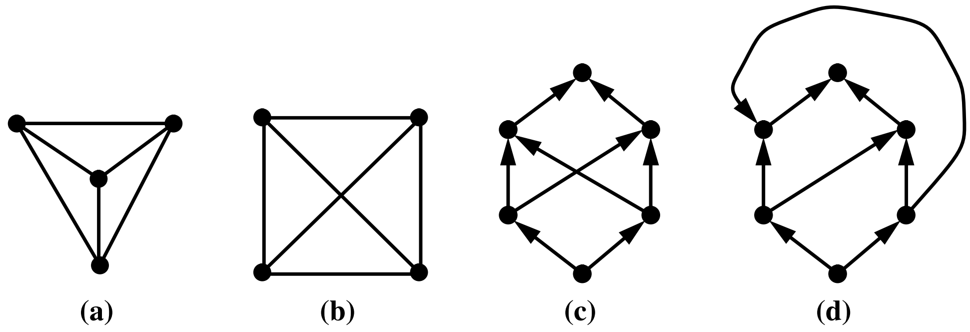
There are infinitely many drawings for a graph. In drawing a graph, we would like to take into account a variety of properties. For example, planarity and the display of symmetries are highly desirable in visualization applications. Also, it is customary to display trees and acyclic digraphs with upward drawings. In general, to avoid wasting valuable space on a page or a computer screen, it is important to keep the area of the drawing small. In this scenario, many graph drawing problems can be formalized as multi-objective optimization problems (e.g., construct a drawing with minimum area and minimum number of crossings), so that trade-offs are inherent in solving them. Typically, it is desirable to maximize the angular resolution and to minimize the other measures.

The following examples illustrate two typical trade-offs in graph drawing problems. Figure 44.1.2(a–b) shows two drawings of $K_4$, the complete graph on four vertices. The drawing of part (a) is planar, while the drawing of part (b) "maximizes symmetries." It can be shown that no drawing of $K_4$ is optimal with respect to both criteria, i.e., the maximum number of symmetries cannot be achieved by a planar drawing. Figure 44.1.2(c–d), shows two drawing of the same acyclic digraph $G$. The drawing of part (c) is upward, while the drawing of part (d) is planar. It can be shown that there is no drawing of $G$ which is both planar and upward.

# 44.2 BOUNDS ON DRAWING PROPERTIES

For various classes of graphs and drawing types, many universal/existential upper and lower bounds for specific drawing properties have been discovered. Such bounds typically exhibit trade-offs between drawing properties. A universal bound applies to all the graphs of a given class. An existential bound applies to infinitely many graphs of the class.

FIGURE 44.1.2
*(a–b) Tradeoff between planarity and symmetry in drawing $K_4$. (c–d) Tradeoff between planarity and upwardness in drawing an acyclic digraph $G$.*




Whenever we give bounds on the area or edge length, we assume that the drawing is constrained by some resolution rule that prevents it from being arbitrarily scaled down (e.g., requiring a grid drawing, or a minimum unit distance between any two vertices).

## BOUNDS ON THE AREA

Table 44.2.1 summarizes selected universal upper bounds and existential lower bounds on the area of drawings of graphs. In the table, $a$ is an arbitrary constant $0 \leq a < 1$, and $b$ and $c$ are fixed constants $1 < b < c$. The abbreviations ***PSL*** and ***PSLg*** are used to abbreviate "planar straight-line" "planar straight-line grid."

In general, the effect of bends on the area requirement is dual. On one hand, bends occupy space and hence negatively affect the area. On the other hand, bends may help in routing edges without using additional space.

The following comments apply to Table 44.2.1, where specific rows of the table are indicated within parentheses. Linear or almost-linear bounds on the area can be achieved for trees (1–7). See Table 44.2.4 for trade-offs between area and aspect-ratio in drawings of trees. Planar graphs admit planar drawings with quadratic area (8–13). However, the area requirement of planar straight-line drawings may be exponential if high angular resolution is also desired (9). Almost linear area instead can be achieved in nonplanar drawings of planar graphs (13), which have applications to VLSI circuits. Upward planar drawings provide an interesting trade-off between area and the total number of bends (14–16). Indeed, unless the digraph is reduced (15), the area can become exponential if a straight-line drawing is required (14). A quadratic area bound is achieved only at the expense of a linear number of bends (16).

## BOUNDS ON THE ANGULAR RESOLUTION

Table 44.2.2 summarizes selected universal lower bounds and existential upper

TABLE 44.2.1 Universal upper bounds and existential lower bounds on the area of drawings of graphs.

| | CLASS OF GRAPHS | DRAWING TYPE | AREA | |
|---|---|---|---|---|
| 1 | rooted tree | upward PSLg | $\Omega(n)$ | $O(n\log n)$ |
| 2 | rooted tree | strictly upward PSLg | $\Omega(n\log n)$ | $O(n\log n)$ |
| 3 | degree-$O(n^a)$ rooted tree | upward planar polyline grid | $\Omega(n)$ | $O(n)$ |
| 4 | binary tree | upward planar orthog grid | $\Omega(n\log\log n)$ | $O(n\log\log n)$ |
| 5 | tree | PSLg | $\Omega(n)$ | $O(n\log n)$ |
| 6 | degree-$O(n^a)$ tree | planar polyline grid | $\Omega(n)$ | $O(n)$ |
| 7 | degree-4 tree | planar orthog grid | $\Omega(n)$ | $O(n)$ |
| 8 | planar graph | planar polyline grid | $\Omega(n^2)$ | $O(n^2)$ |
| 9 | planar graph | PSL | $\Omega(c^{\rho n})$ | |
| 10 | planar graph | PSLg | $\Omega(n^2)$ | $O(n^2)$ |
| 11 | triconnected planar graph | PSL convex grid | $\Omega(n^2)$ | $O(n^2)$ |
| 12 | planar graph | planar orthog grid | $\Omega(n^2)$ | $O(n^2)$ |
| 13 | planar degree-4 graph | orthog grid | $\Omega(n\log n)$ | $O(n\log^2 n)$ |
| 14 | upward planar digraph | upward PSLg | $O(b^n)$ | $\Omega(c^n)$ |
| 15 | reduced planar $st$-digraph | upward PSLg dominance | $\Omega(n^2)$ | $O(n^2)$ |
| 16 | upward planar digraph | upward planar grid polyline | $O(n^2)$ | $\Omega(n^2)$ |
| 17 | general graph | polyline grid | $\Omega(n+\chi)$ | $O((n+\chi)^2)$ |

bounds on the angular resolution of drawings of graphs. Here $c$ a fixed constant $c > 1$.

TABLE 44.2.2 Universal lower bounds and existential upper bounds on the angular resolution of drawings of graphs.

| CLASS OF GRAPHS | DRAWING TYPE | ANGULAR RESOLUTION | |
|---|---|---|---|
| general graph | straight-line | $\Omega(\frac{1}{d^2})$ | $O(\frac{\log d}{d^2})$ |
| planar graph | straight-line | $\Omega(\frac{1}{d})$ | $O(\frac{1}{d})$ |
| planar graph | planar straight-line | $\Omega(\frac{1}{c^d})$ | $O(\sqrt{\frac{\log d}{d^3}})$ |

## BOUNDS ON THE NUMBER OF BENDS

Table 44.2.3 summarizes selected universal upper bounds and existential lower bounds on the total and maximum number of bends in orthogonal drawings. Some bounds are stated for $n \geq 5$ or $n \geq 7$ because the maximum number of bends is at least 2 for $K_4$ and at least 3 for the skeleton graph of an octahedron, in any planar orthogonal drawing.

TABLE 44.2.3 Orthogonal drawings: universal upper bounds and existential lower bounds on the total and maximum number of bends. Notes: † $n \geq 7$; ‡ $n \geq 5$.

| CLASS OF GRAPHS | DRAWING TYPE | TOTAL # BENDS | | MAX # BENDS | |
|---|---|---|---|---|---|
| deg-4 † | orthog | $\geq n$ | $\leq 2n+2$ | $\geq 2$ | $\leq 2$ |
| planar deg-4 † | orthog planar | $\geq 2n-2$ | $\leq 2n+2$ | $\geq 2$ | $\leq 2$ |
| embedded deg-4 | orthog planar | $\geq 2n-2$ | $\leq \frac{12}{5}n+2$ | $\geq 3$ | $\leq 3$ |
| biconn embedded deg-4 | orthog planar | $\geq 2n-2$ | $\leq 2n+2$ | $\geq 3$ | $\leq 3$ |
| triconn embedded deg-4 | orthog planar | $\geq \frac{4}{3}(n-1)+2$ | $\leq \frac{3}{2}n+4$ | $\geq 2$ | $\leq 2$ |
| embedded deg-3 ‡ | orthog planar | $\geq \frac{1}{2}n+1$ | $\leq \frac{1}{2}n+1$ | $\geq 1$ | $\leq 1$ |

## TRADE-OFF BETWEEN AREA AND ASPECT-RATIO

The ability to construct area-efficient drawings is essential in practical visualization applications, where screen space is at a premium. However, achieving small area is not enough: e.g., a drawing with high aspect-ratio may not be conveniently placed on a workstation screen, even if it has modest area. Hence, it is important to keep the aspect-ratio small. Ideally, one would like to obtain small area for any given aspect-ratio in a wide range. This would provide graphical user interfaces with the flexibility of fitting drawings in arbitrarily shaped windows.

A variety of trade-offs for the area and aspect-ratio arise even when drawing graphs with a simple structure, such as trees. Table 44.2.4 summarizes selected universal bounds that can be simultaneously achieved on the area and the aspect-ratio of various types of drawings of trees.

TABLE 44.2.4 Universal upper bounds that can be simultaneously achieved for the area and aspect-ratio in drawings of trees. $a$ is an arbitrary constant $0 \leq a < 1$.

| CLASS OF GRAPHS | DRAWING TYPE | AREA | ASPECT-RATIO |
|---|---|---|---|
| rooted tree | upward PSL layered grid | $O(n^2)$ | $O(1)$ |
| rooted tree | upward PSL grid | $O(n \log n)$ | $O(n/\log n)$ |
| rooted deg-$O(1)$ tree | upward planar polyline grid | $O(n)$ | $O(n^a)$ |
| binary tree | upward planar orthog grid | $O(n \log\log n)$ | $O(n \log\log n/\log^2 n)$ |
| deg-4 tree | orthog grid | $O(n)$ | $O(1)$ |
| deg-4 tree | orthog grid, leaves on convex hull | $O(n \log n)$ | $O(1)$ |

While upward planar straight-line drawings are the most natural way of visualizing rooted trees, the existing drawing techniques are unsatisfactory with respect to either the area requirement or the aspect ratio. The situation is similar for orthogonal drawings. Regarding polyline drawings, linear area can be achieved with a prescribed aspect ratio. However, experiments show that this is done at the expense of a somehow aesthetically unappealing drawing.

For non-upward drawings of trees, linear area and optimal aspect ratio are possible for planar orthogonal drawings, and a small (logarithmic) amount of extra area is needed if the leaves are constrained to be on the convex hull of the drawing (e.g., pins on the boundary of a VLSI circuit). However, the non-upward drawing methods do not seem to yield aesthetically pleasing drawings, and are suited more for VLSI layout than for visualization applications.

## TRADE-OFF BETWEEN AREA AND ANGULAR RESOLUTION

Table 44.2.5 summarizes selected universal bounds that can be simultaneously achieved on the area and the angular resolution of drawings of graphs.

TABLE 44.2.5 Universal asymptotic upper bounds for the area and lower bounds for the angular resolution that can be simultaneously achieved in drawings of graphs. $b$ and $c$ are fixed constants, $b > 1$ and $c > 1$.

| CLASS OF GRAPHS | DRAWING TYPE | AREA | ANGULAR RESOLUTION |
|---|---|---|---|
| planar graph | straight-line | $O(d^6 n)$ | $\Omega(\frac{1}{d^2})$ |
| planar graph | straight-line | $O(d^3 n)$ | $\Omega(\frac{1}{d})$ |
| planar graph | planar straight-line grid | $O(n^2)$ | $\Omega(\frac{1}{n^2})$ |
| planar graph | planar straight-line | $O(b^n)$ | $\Omega(\frac{1}{c^d})$ |
| planar graph | planar polyline grid | $O(n^2)$ | $\Omega(\frac{1}{d})$ |

Universal lower bounds on the angular resolution exist that depend only on the degree of the graph. Also, substantially better bounds can be achieved by drawing a planar graph with bends or in a nonplanar way.

## OPEN PROBLEMS

1. Determine the area requirement of (upward) planar straight-line drawings of trees. There is currently an $O(\log n)$ gap between the known upper and lower bounds (Table 44.2.1).

2. Determine the area requirement of orthogonal (or, more generally, polyline) nonplanar drawings of planar graphs. There is currently an $O(\log n)$ gap between the known upper and lower bounds (Table 44.2.1).

3. Close the wide gap between the $\Omega(\frac{1}{d^2})$ universal lower bound and the $O(\frac{\log d}{d^2})$ existential upper bound on the angular resolution of straight-line drawings of general graphs (Table 44.2.2).

4. Close the gap between the $\Omega(\frac{1}{c^d})$ universal lower bound and the $O(\sqrt{\frac{\log d}{d^3}})$ existential upper bound on the angular resolution of planar straight-line drawings of planar graphs (Table 44.2.2).

5. Determine the best possible aspect-ratio and area that can be simultaneously achieved for (upward) planar straight-line and orthogonal drawings of trees (Table 44.2.4).

# 44.3 COMPLEXITY OF GRAPH DRAWING PROBLEMS

Tables 44.3.1–44.3.3 summarize selected results on the time complexity of some fundamental graph drawing problems.

TABLE 44.3.1 The time complexity of some fundamental graph drawing problems: general graphs and digraphs.

| CLASS OF GRAPHS | PROBLEM | TIME COMPLEXITY | |
|---|---|---|---|
| general graph | minimize crossings | NP-hard | |
| 2-layered graph | minimize crossings in layered drawing with preassigned order on one layer | NP-hard | |
| general graph | maximum planar subgraph | NP-hard | |
| general graph | planarity testing and computing a planar embedding | $\Omega(n)$ | $O(n)$ |
| general graph | maximal planar subgraph | $\Omega(n+m)$ | $O(n+m)$ |
| general digraph | upward planarity testing | NP-hard | |
| embedded digraph | upward planarity testing | $\Omega(n)$ | $O(n^2)$ |
| single-source digraph | upward planarity testing | $\Omega(n)$ | $O(n)$ |
| general graph | draw as the intersection graph of a set of unit diameter disks in the plane | NP-hard | |

It is interesting that apparently similar problems exhibit very different time complexities. For example, while planarity testing can be done in linear time, upward planarity testing is NP-hard. Note that, as illustrated in Figure 44.1.2(c–d), planarity and acyclicity are necessary but not sufficient conditions for upward planarity.

While many efficient algorithms exists for constructing drawings of trees and planar graphs with good universal area bounds, exact area minimization for most types of drawings is NP-hard, even for trees.

## OPEN PROBLEMS

1. Reduce the time complexity of upward planarity testing for embedded digraphs (currently $O(n^2)$), or prove a superlinear lower bound (Table 44.3.1).

TABLE 44.3.2 The time complexity of some fundamental graph drawing problems: planar graphs and digraphs.

| CLASS OF GRAPHS | PROBLEM | TIME COMPLEXITY | |
|---|---|---|---|
| planar graph | planar straight-line drawing with prescribed edge lengths | NP-hard | |
| planar graph | planar straight-line drawing with maximum angular resolution | NP-hard | |
| embedded graph | test the existence of a planar straight-line drawing with prescribed angles between pairs of consecutive edges incident on a vertex | NP-hard | |
| maximal planar graph | test the existence of a planar straight-line drawing with prescribed angles between pairs of consecutive edges incident on a vertex | $\Omega(n)$ | $O(n)$ |
| planar graph | planar straight-line grid drawing with $O(n^2)$ area and $O(1/n^2)$ angular resolution | $\Omega(n)$ | $O(n)$ |
| planar graph | planar polyline drawing with $O(n^2)$ area, $O(n)$ bends, and $O(1/d)$ angular resolutions | $\Omega(n)$ | $O(n)$ |
| triconn planar graph | planar straight-line convex grid drawing with $O(n^2)$ area and $O(1/n^2)$ angular resolution | $\Omega(n)$ | $O(n)$ |
| triconn planar graph | planar straight-line strictly convex drawing | $\Omega(n)$ | $O(n)$ |
| reduced planar $st$-digraph | upward planar grid straight-line dominance drawing with minimum area | $\Omega(n)$ | $O(n)$ |
| upward planar digraph | upward planar polyline grid drawing witn $O(n^2)$ area and $O(n)$ bends | $\Omega(n)$ | $O(n)$ |
| planar deg-4 graph | planar orthogonal grid drawing with minimum number of bends | NP-hard | |
| planar deg-3 graph | planar orthogonal grid drawing with minimum number of bends and $O(n^2)$ area | $\Omega(n)$ | $O(n^5 \log n)$ |
| embedded deg-4 graph | planar orthogonal grid drawing with minimum number of bends and $O(n^2)$ area | $\Omega(n)$ | $O(n^{7/4} \log n)$ |
| planar deg-4 graph | planar orthogonal grid drawing with $O(n^2)$ area and $O(n)$ bends | $\Omega(n)$ | $O(n)$ |

2. Reduce the time complexity of bend minimization for planar orthogonal drawings of embedded graphs (currently $O(n^{7/4} \log n)$), or prove a superlinear lower bound (Table 44.3.2).

TABLE 44.3.3 The time complexity of some fundamental graph drawing problems: trees. $k$ is a fixed constant $k \geq 1$.

| CLASS OF GRAPHS | PROBLEM | TIME COMPLEXITY | |
|---|---|---|---|
| tree | draw as the Euclidean minimum spanning tree of a set of points in the plane | NP-hard | |
| degree-4 tree | minimize area in planar orthogonal grid drawing | NP-hard | |
| degree-4 tree | minimize total/maximum edge length in planar orthogonal grid drawing | NP-hard | |
| rooted tree | minimize area in a planar straight-line upward layered grid drawing that displays symmetries and isomorphisms of subtrees | NP-hard | |
| rooted tree | minimize area in a planar straight-line upward layered drawing that displays symmetries and isomorphisms of subtrees | $\Omega(n)$ | $O(n^k)$ |
| binary tree | minimize area in hv-drawing | $\Omega(n)$ | $O(n\sqrt{n \log n})$ |
| rooted tree | planar straight-line upward layered grid drawing with $O(n^2)$ area | $\Omega(n)$ | $O(n)$ |
| rooted tree | planar polyline upward grid drawing with $O(n)$ area | $\Omega(n)$ | $O(n)$ |

## 44.4 EXAMPLE OF GRAPH DRAWING ALGORITHM

In this section we outline the algorithm by the author [Tam87] for computing a planar orthogonal grid drawing with minimum number of bends and $O(n^2)$ area of an embedded degree-4 graph $G$ (see Table 44.3.2). This algorithm is the core of a practical drawing algorithm for general graphs (see Section 44.5 and Figure 44.4.1(d)).

The algorithm consists of two main phases:

1. computation of an *orthogonal shape* for $G$, where only the bends and the angles of the orthogonal drawing are defined;
2. assignment of integer lengths to the segments of the orthogonal shape.

Phase 1 uses a transformation into a network flow problem (Figure 44.4.1(a–c)), where each unit of flow is associated with a right angle in the orthogonal drawing. Hence, angles are viewed as a commodity that is produced by the vertices, transported across faces by the edges through their bends, and eventually consumed by the faces.

From the embedded graph $G$ we construct a flow network $N$ as follows. The nodes of network $N$ are the vertices and faces of $G$. Let $\deg(f)$ denote the number of edges of the circuit bounding face $f$. Each vertex $v$ supplies $\sigma(v) = 4$ units of flow, and each face $f$ consumes $\tau(f)$ units of flow, where

$$\tau(f) = \begin{cases} 2\deg(f) - 4 & \text{if } f \text{ is an internal face} \\ 2\deg(f) + 4 & \text{if } f \text{ is the external face} \end{cases}$$

By Euler's formula, $\sum_v \sigma(v) = \sum_f \tau(f)$, i.e., the total supply is equal to the total consumption.

Network $N$ has two types of arcs:

- arcs of the type $(v, f)$, where $f$ is a face incident on vertex $v$; the flow in $(v, f)$ represents the angle at vertex $v$ in face $f$, and has lower bound 1, upper bound 4, and cost 0;
- arcs of the type $(f, g)$, where face $f$ shares an edge $e$ with face $g$; the flow in $(f, g)$ represents the number of bends along edge $e$ with the right angle inside face $f$, and has lower bound 0, upper bound $+\infty$, and cost 1.

The conservation of flow at the vertices expresses the fact that the sum of the angles around a vertex is equal to $2\pi$. The conservation of flow at the faces expresses that fact that the sum of the angles at the vertices and bends of an internal face is equal to $\pi(p-2)$, where $p$ is the number of such angles. For the external face, the above sum is equal to $\pi(p+2)$.

It can be shown that every feasible flow $\phi$ in network $N$ corresponds to an admissible orthogonal shape for graph $G$, whose number of bends is equal to the cost of flow $\phi$. Hence, an orthogonal shape for $G$ with the minimum number of bends can be computed from a minimum cost flow in $G$. This can be done in $O(n^2 \log n)$ time with standard flow-augmentation methods.

Phase 2 uses a simple compaction strategy derived from VLSI layout, where the lengths of the horizontal and vertical segments are computed independently after a preliminary refinement of the orthogonal shape that decomposes each face into into rectangles.

## 44.5 TECHNIQUES FOR DRAWING GRAPHS

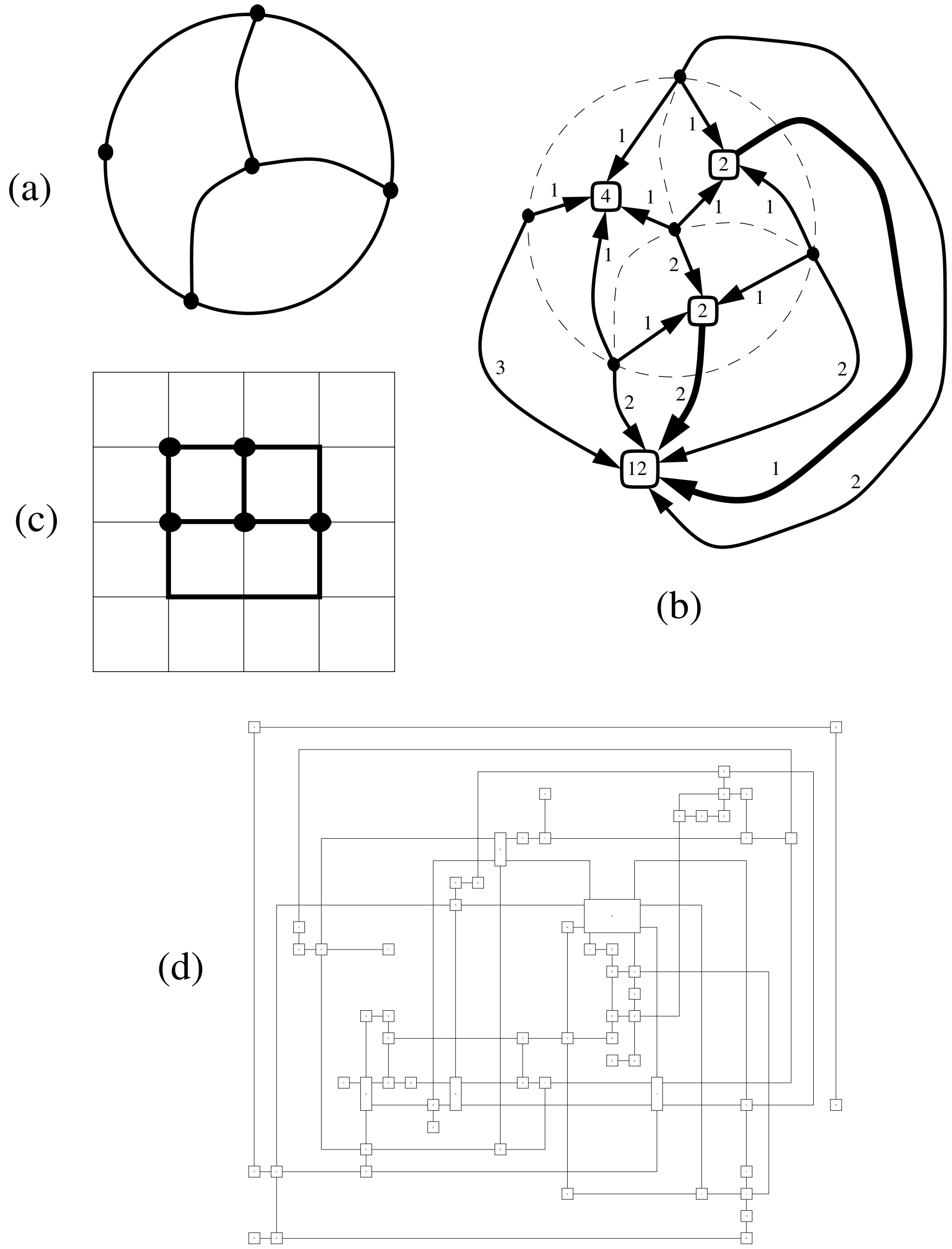
In this section we outline some of the most successful techniques that have been devised for drawing general graphs.

### PLANARIZATION

The planarization approach is motivated by the availability of many efficient and well-analyzed drawing algorithms for planar graphs (See Table 44.3.2). If the graph is nonplanar, it is transformed into a planar graph by means of a preliminary planarization step that replaces each crossing with a fictitious vertex. Finding the minimum number of crossings or a maximum planar subgraph are NP-hard problems. Hence, existing planarization algorithms use heuristics.

The best available heuristic for the maximum planar subgraph problem is described in [JM96]. This method has a solid theoretical foundation in polyhedral combinatorics, and achieves good results in practice.

FIGURE 44.4.1
*(a) Embedded graph $G$. (b) Minimum cost flow in network $N$: the flow is shown next to each arc; arcs with zero flow are omitted; arcs with unit cost are drawn with thick lines; a face $f$ is represented by a box labeled with $\tau(f)$. (c) Planar orthogonal grid drawing of $G$ with minimum number of bends. (d) Orthogonal grid drawing of a nonplanar graph produced by a drawing method for general graphs based on the algorithm of Section 44.4.*



A successful drawing algorithm based on the planarization approach and a bend-minimization method [Tam87] is described in [TDB88] (Figure 44.4.1(d) was generated by this algorithm). It has been widely used in software visualization systems,

### LAYERING

The layering approach for constructing polyline drawings of directed graphs transforms the digraph into a layered digraph and then constructs a layered drawing. A typical algorithm based on the layering approach consists of the following main steps:

1. Assign each vertex to a layer, with the goal of maximizing the number of edges oriented upward.
2. Insert fictitious vertices along the edges crossing layers, such that each edge in the resulting digraph connects vertices in consecutive layers. (The fictitious vertices will be displayed as bends in the final drawing.)
3. Permute the vertices on each layer with the goal of minimizing crossings.
4. Adjust the position of the vertices in each layer with the goal of distributing the vertices uniformly and minimizing the number of bends.

Most of the subproblems involved in the various steps are NP-hard, hence heuristics must be used. The layering approach was pioneered by Sugiyama et al. [STT81]. The most notable developments of this technique are due to Gansner et al [GNV88, GKNV93].

### PHYSICAL SIMULATION

This approach uses a physical model where the vertices and edges of the graph are viewed as objects subject to various forces. Starting from an initial random configuration, the physical system evolves into a final configuration of minimum energy, which yields the drawing. Rather than solving a system of differential equations, the evolution of the system is usually simulated using numerical methods (e.g., at each step, the forces are computed and corresponding incremental displacements of the vertices are performed). Drawing algorithms based on the physical simulation approach are often able to detect and display symmetries in the graph. However, their running time is typically high.

The physical simulation approach was pioneered in [Ead84, KS80]. Recent sophisticated developments include [DH, FR91].

## 44.6 RECENT RESEARCH TRENDS

In this section, we overview selected areas of graph drawing that have recently attracted increasing attention.

## THREE-DIMENSIONAL DRAWINGS

Recent advances in hardware and software technology for computer graphics open the possibility of displaying three-dimensional (3D) visualizations on a variety of low-cost workstations. Previous research on 3D graph drawing has focused on the development of visualization systems.

Much work needs to be done on the theoretical foundations of 3D graph drawing. Recent progress has been reported in [CGT96, CELR95, LD95].

## PROXIMITY AND VISIBILITY DRAWINGS

Geometric representations of graphs by means of the vertical visibility relation among horizontal segments are well understood (see, e.g., [TT86]). They have applications to motion planning and VLSI layout. Recent progress on visibility drawings has been reported in [AGW96].

Proximity drawings include Gabriel, relative neighborhood, Delaunay, sphere of influence, and minimum spanning drawings. Increasing attention has been devoted to the problem of characterizing the classes of graphs that admit various types of proximity drawings. A survey of this area appears in [DLL95].

## DECLARATIVE METHODS

Research in graph drawing has traditionally focused on algorithmic methods, where the drawing of the graph is generated according to a prespecified set of aesthetic criteria (such as planarity or area minimization) that are embodied in an algorithm. Although the algorithmic approach is computationally efficient, it does not naturally support constraints, i.e., requirements that the user may want to impose on the drawing of a specific graph (e.g., clustering or aligning a given set of vertices). Previous work has shown that only a rather limited constraint satisfaction capability can be added to existing drawing algorithms (see, e.g., [TDB88]).

Recently, several attempts have been made at developing languages for the specification of constraints and at devising techniques for graph drawing based on the resolution of systems of constraints (see, e.g., [Kam89]). A visual approach to graph drawing, where the layout of a graph is pictorially specified "by example," is proposed by Cruz and Garg [CG95]. Recent work by Eades and Lin [LE95] also attempts at combining algorithmic and declarative methods. Brandenburg presents a comprehensive approach to graph drawing based on graph grammars [Bra95].

Related work includes the physical simulation approach (Section 44.5) and genetic algorithms [KMS94].

## DYNAMIC GRAPH DRAWING

Many graphic interfaces provide graph editors that allow the user to interactively modify the drawing of a graph. After an editing session, it would be useful to have an algorithm that "beautifies" the drawing with smooth modifications that preserve the user's "mental map" of the diagram [ELMS91].

This scenario motivates the development of dynamic algorithms that perform incremental changes to the drawing of a graph subject to a sequence of update operations, such as insertion and deletions of vertices and edges. Trade-offs between running time, optimization of the drawing properties, and preservation of the mental map are typical issues to be addressed in dynamic graph drawing.

Most of the existing results on dynamic graph drawing are limited to planar graphs (see, e.g., [CDTT95]).

### EXPERIMENTATION

Many graph drawing algorithms have been implemented and used in practical applications. Most papers show sample outputs, and some also provide limited experimental results on small test suites. However, in order to evaluate the practical performance of a graph drawing algorithm in visualization applications, it is essential to perform extensive experimentations with input graphs derived from the application domain.

The performance of four planar straight-line drawing algorithms on 10,000 randomly generated maximal planar graphs is compared by Jones et al. [JEM+91].

Himsolt [Him95a] presents a comparative study of twelve graph drawings algorithms based on various approaches. The experiments are conducted on 100 sample graphs with the graph drawing system *GraphEd* [Him95b]. Many examples of drawings constructed by the algorithms are shown, and various objective and subjective evaluations on the aesthetic quality of the drawings produced are given.

Di Battista et al. [DGL+95] report on an extensive experimental study comparing three orthogonal drawing algorithms based on the planarization approach. The test data are 11,582 graphs, ranging from 10 to 100 vertices, which have been generated from a core set of 112 graphs used in "real-life" software engineering and database applications. The experiments are conducted with *Diagram Server* [DGST90, DLV95] (a graph drawing system based on a client-server architecture) and provide a detailed quantitative evaluation of the performance of the three algorithms.

The observed practical behavior of the three algorithms was consistent with their theoretical properties. Namely,the algorithm of [TDB88], based on the planarization approach and a bend-minimization method [Tam87] (see Section 44.4 and Figure 44.4.1(d)), clearly outperformed the other two algorithms, based on linear-time heuristic methods, for most quality measures (including crossings, area, and bends) but was considerably slower.

## 44.7 SOURCES AND RELATED MATERIAL

A comprehensive bibliography on graph drawing algorithms [DETT94] cites more than 300 papers written before 1993. Most papers on graph drawing are cited in `geom.bib`, the computational geometry BibTEX bibliography available from `ftp://cs.usask.ca/pub/geometry/` (search for keyword "graph drawing"). Surveys on various aspects of graph drawing appear in [DLL95, EL89, GT95, HK95, Riv89, Riv93, SSV95, Tam90a, Tam90b]. The author maintains a WWW page (`http://www.cs.brown.edu/people/rt/gd.html`) with pointers to graph drawing resources on the Web.

The proceedings of the annual Symposium on Graph Drawing are published by Springer-Verlag in the LNCS series [TT95, Bra96]. Special issues dedicated to graph drawing appeared or will appear in *Algorithmica* (edited by G. Di Battista and R. Tamassia, vol. 16, no. 1, 1996), *Computational Geometry: Theory and Applications* (edited by G. Di Battista and R. Tamassia, to appear), and the *Journal of Visual Languages and Computing* (edited by I. F. Cruz and P. Eades, vol. 6, no. 3, 1995).

## RELATED CHAPTERS

Chapter 25: Visibility

## REFERENCES


[AGW96] H. Alt, M. Godau and S. Whitesides, *Universal 3-Dimensional Visibility Representations for Graphs*, In F.J. Brandenburg, editor, Graph Drawing (Proc. GD '95), vol. 1027 of *Lecture Notes in Computer Science*, pp. 8–19. Springer-Verlag, 1996.

[Bra95] F.J. Brandenburg, *Designing Graph Drawings by Layout Graph Grammars*, In R. Tamassia and I.G. Tollis, editors, Graph Drawing (Proc. GD '94), vol. 894 of *Lecture Notes in Computer Science*, pp. 416–427. Springer-Verlag, 1995.

[Bra96] F.J. Brandenburg, editor, *Graph Drawing (Proc. GD '95)*, vol. 1027 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996.

[CDTT95] R.F. Cohen, G. Di Battista, R. Tamassia and I.G. Tollis, *Dynamic Graph Drawings: Trees, Series-Parallel Digraphs, and Planar ST-Digraphs*, SIAM J. Comput., 24, no. 5, 970–1001, 1995.

[CELR95] R.F. Cohen, P. Eades, T. Lin and F. Ruskey, *Three-Dimensional Graph Drawing*, In R. Tamassia and I.G. Tollis, editors, Graph Drawing (Proc. GD '94), vol. 894 of *Lecture Notes in Computer Science*, pp. 1–11. Springer-Verlag, 1995.

[CG95] I.F. Cruz and A. Garg, *Drawing Graphs by Example Efficiently: Trees and Planar Acyclic Digraphs*, In R. Tamassia and I.G. Tollis, editors, Graph Drawing (Proc. GD '94), vol. 894 of *Lecture Notes in Computer Science*, pp. 404–415. Springer-Verlag, 1995.

[CGT96] M. Chrobak, M.T. Goodrich and R. Tamassia, *Convex Drawings of Graphs in Two and Three Dimensions*, In Proc. 12th Annu. ACM Sympos. Comput. Geom., pp. 319–328, 1996.

[DETT94] G. Di Battista, P. Eades, R. Tamassia and I.G. Tollis, *Algorithms for drawing graphs: an annotated bibliography*, Comput. Geom. Theory Appl., 4, 235–282, 1994.

[DGL+95] G. Di Battista *et al.*, *An Experimental Comparison of Three Graph Drawing Algorithms*, In Proc. 11th Annu. ACM Sympos. Comput. Geom., pp. 306–315, 1995.

[DGST90] G. Di Battista, A. Giammarco, G. Santucci and R. Tamassia, *The Architecture of Diagram Server*, In Proc. IEEE Workshop on Visual Languages (VL'90), pp. 60–65, 1990.

[DH] R. Davidson and D. Harel, *Drawing Graphs Nicely Using Simulated Annealing*, Commun. ACM, To appear.

[DLL95] G. Di Battista, W. Lenhart and G. Liotta, *Proximity Drawability: a Survey*, In R. Tamassia and I.G. Tollis, editors, Graph Drawing (Proc. GD '94), vol. 894 of *Lecture Notes in Computer Science*, pp. 328–339. Springer-Verlag, 1995.

[DLV95] G. Di Battista, G. Liotta and F. Vargiu, *Diagram Server*, J. Visual Languages and Computing, 6, no. 3, 275–298, 1995, (special issue on Graph Visualization, edited by I. F. Cruz and P. Eades).

[Ead84] P. Eades, *A Heuristic for Graph Drawing*, Congr. Numer., 42, 149–160, 1984.

[EL89] P. Eades and X. Lin, *How to Draw a Directed Graph*, In Proc. IEEE Workshop on Visual Languages (VL'89), pp. 13–17, 1989.

[ELMS91] P. Eades, W. Lai, K. Misue and K. Sugiyama, *Preserving the Mental Map of a Diagram*, In Proceedings of Compugraphics 91, pp. 24–33, 1991.

[FR91] T. Fruchterman and E. Reingold, *Graph Drawing by Force-Directed Placement*, Softw. – Pract. Exp., 21, no. 11, 1129–1164, 1991.

[GKNV93] E.R. Gansner, E. Koutsofios, S.C. North and K.P. Vo, *A Technique for Drawing Directed Graphs*, IEEE Trans. Softw. Eng., 19, 214–230, 1993.

[GNV88] E.R. Gansner, S.C. North and K.P. Vo, *DAG – A Program that Draws Directed Graphs*, Softw. – Pract. Exp., 18, no. 11, 1047–1062, 1988.

[GT95] A. Garg and R. Tamassia, *Upward Planarity Testing*, Order, 12, 109–133, 1995.

[Him95a] M. Himsolt, *Comparing and Evaluating Layout Algorithms within GraphEd*, J. Visual Languages and Computing, 6, no. 3, 1995, (special issue on Graph Visualization, edited by I. F. Cruz and P. Eades).

[Him95b] M. Himsolt, *GraphEd: a Graphical Platform for the Implementation of Graph Algorithms*, In R. Tamassia and I.G. Tollis, editors, Graph Drawing (Proc. GD '94), vol. 894 of *Lecture Notes in Computer Science*, pp. 182–193. Springer-Verlag, 1995.

[HK95] X. He and M.-Y. Kao, *Regular Edge Labelings and Drawings of Planar Graphs*, In R. Tamassia and I.G. Tollis, editors, Graph Drawing (Proc. GD '94), vol. 894 of *Lecture Notes in Computer Science*, pp. 96–103. Springer-Verlag, 1995.

[JEM+91] S. Jones *et al.*, *A Note on Planar Graph Drawing Algorithms*, Technical Report 216, Department of Computer Science, University of Queensland, 1991.

[JM96] M. Jünger and P. Mutzel, *Maximum Planar Subgraphs and Nice Embeddings: Practical Layout Tools*, Algorithmica, 16, no. 1, 1996, (special issue on Graph Drawing, edited by G. Di Battista and R. Tamassia).

[Kam89] T. Kamada, *Visualizing Abstract Objects and Relations*. World Scientific Series in Computer Science, 1989.

[KMS94] C. Kosak, J. Marks and S. Shieber, *Automating the Layout of Network Diagrams with Specified Visual Organization*, IEEE Trans. Syst. Man Cybern., 24, no. 3, 440–454, 1994.

[KS80] J.B. Kruskal and J.B. Seery, *Designing Network Diagrams*, In Proc. First General Conference on Social Graphics, pp. 22–50. U. S. Department of the Census, 1980.

[LD95] G. Liotta and G. Di Battista, *Computing Proximity Drawings of Trees in the 3-Dimemsional Space*, In Proc. 4th Workshop Algorithms Data Struct., vol. 955 of *Lecture Notes in Computer Science*, pp. 239–250. Springer-Verlag, 1995.

[LE95] T. Lin and P. Eades, *Integration of Declarative and Algorithmic Approaches for Layout Creation*, In R. Tamassia and I.G. Tollis, editors, Graph Drawing (Proc. GD '94), vol. 894 of *Lecture Notes in Computer Science*, pp. 376–387. Springer-Verlag, 1995.

[Riv89] I. Rival, *Graphical Data Structures for Ordered Sets*, In I. Rival, editor, Algorithms and Order, pp. 3–31. Kluwer Academic Publishers, 1989.

[Riv93] I. Rival, *Reading, Drawing, and Order*, In I.G. Rosenberg and G. Sabidussi, editors, Algebras and Orders, pp. 359–404. Kluwer Academic Publishers, 1993.

[SSV95] F. Shahrokhi, L.A. Székely and I. Vrt'o, *Crossing Numbers of Graphs, Lower Bound Techniques and Algorithms: a Survey*, In R. Tamassia and I.G. Tollis, editors, Graph Drawing (Proc. GD '94), vol. 894 of *Lecture Notes in Computer Science*, pp. 131–142. Springer-Verlag, 1995.

[STT81] K. Sugiyama, S. Tagawa and M. Toda, *Methods for Visual Understanding of Hierarchical Systems*, IEEE Trans. Syst. Man Cybern., SMC-11, no. 2, 109–125, 1981.

[Tam87] R. Tamassia, *On Embedding a Graph in the Grid with the Minimum Number of Bends*, SIAM J. Comput., 16, no. 3, 421–444, 1987.

[Tam90a] R. Tamassia, *Drawing Algorithms for Planar st-Graphs*, Australasian Journal of Combinatorics, 2, 217–235, 1990.

[Tam90b] R. Tamassia, *Planar Orthogonal Drawings of Graphs*, In Proc. IEEE Internat. Sympos. on Circuits and Systems, 1990.

[TDB88] R. Tamassia, G. Di Battista and C. Batini, *Automatic Graph Drawing and Readability of Diagrams*, IEEE Trans. Syst. Man Cybern., SMC-18, no. 1, 61–79, 1988.

[TT86] R. Tamassia and I.G. Tollis, *A Unified Approach to Visibility Representations of Planar Graphs*, Discrete Comput. Geom., 1, no. 4, 321–341, 1986.

[TT95] R. Tamassia and I.G. Tollis, editors, *Graph Drawing (Proc. GD '94)*, vol. 894 of *Lecture Notes in Computer Science*. Springer-Verlag, 1995.