Algorithms For Drawing Graphs: An Annotated Bibliography

Peter Eades[1]
Roberto Tamassia[2]

Technical Report No. CS-89-09
Revised Version
October 1989

[1] University of Queensland, Dept. of Computer Science, St. Lucia, Queensland 4067 - Australia
[2] Brown University, Dept. of Computer Science, Providence, RI 02912

ALGORITHMS FOR DRAWING GRAPHS: AN ANNOTATED BIBLIOGRAPHY *

Peter Eades [1] and Roberto Tamassia [2]

[1] Department of Computer Science
University of Queensland
St. Lucia, Queensland 4067 - Australia

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

Abstract

Several data presentation problems involve drawing graphs so that they are easy to read and understand. Examples include circuit schematics and diagrams for information systems analysis and design. In this paper we present a bibliographic survey on algorithms whose goal is to produce aesthetically pleasing drawings of graphs. Research on this topic is spread over the broad spectrum of Computer Science. This bibliography constitutes a first attempt to encompass both theoretical and application oriented papers from disparate areas.

(Revised Version — October 1989)

* Research partially supported by Cadre Technologies, Inc..

1. Introduction

A number of data presentation problems involve the drawing of a graph on a 2-dimensional surface. Examples include circuit schematics, algorithm animation, and diagrams for information systems analysis and design. In this paper we present a bibliographic survey on algorithms whose goal is to produce clear and readable drawings of graphs.

Various *graphic standards* have been proposed for the representation of graphs in the plane. Usually, the vertices are represented by symbols such as circles or boxes, and each edge (u,v) is represented by a simple open curve between the symbols associated with the vertices u and v. A drawing such that each edge is represented by a polygonal chain is a *polyline* drawing (see Fig. 1.a). There are two common special cases of this standard. A *straight-line* drawing maps each edge into a straight-line segment (see Fig. 2). This standard is commonly adopted in graph theory texts. An *orthogonal* drawing maps each edge into a chain of horizontal and vertical segments (see Fig. 3). Entity Relationship diagrams in data base design are usually drawn according to this standard. Note that polyline drawings can be modified to give drawings with nicely curved edges (see Fig. 1.b). A drawing is *planar* if no two edges intersect. A polyline drawing is a *grid* drawing if the vertices and the bends of the edges have integer coordinates.

A *graph drawing algorithm* reads as input a combinatorial description of a graph G, and produces as output a drawing of G according to a given graphic standard.

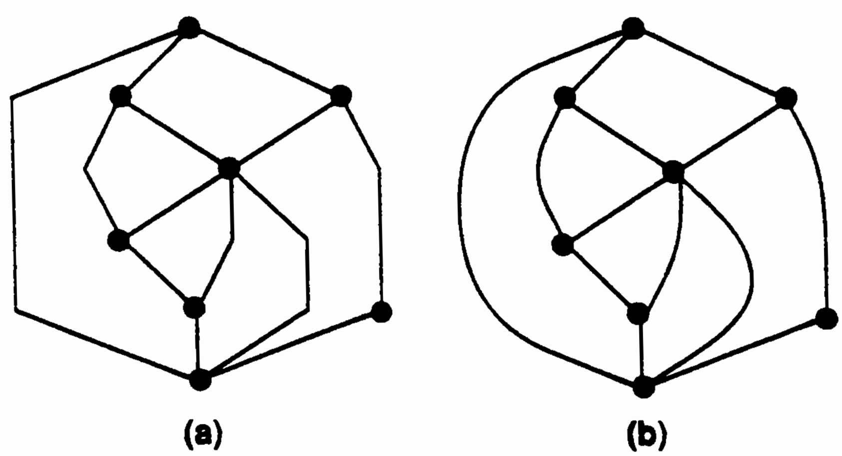

Figure 1 (a) Polyline drawing. (b) Drawing with curved edges.

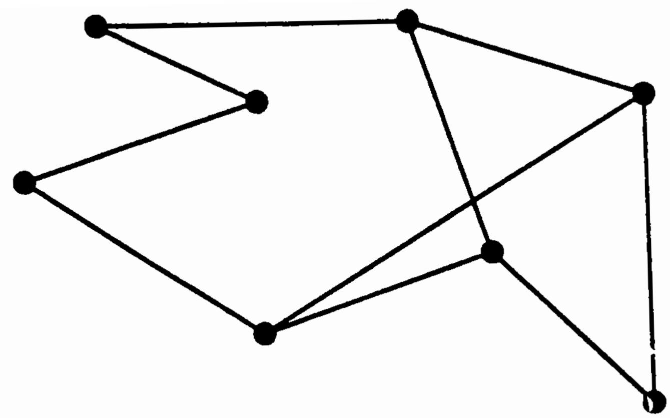

Figure 2 Straight-line drawing.

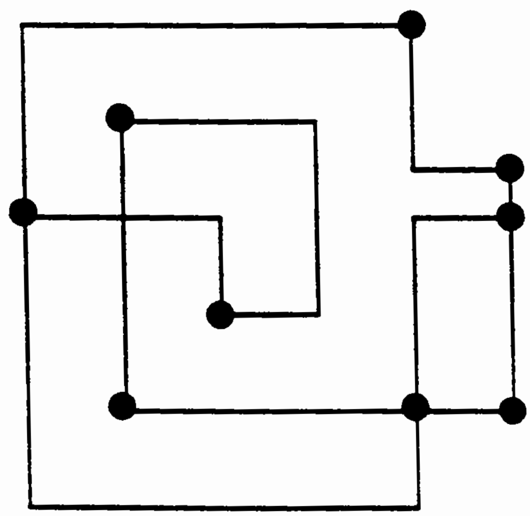

Figure 3 Orthogonal drawing.

Within a graphic standard, a graph has infinitely many different drawings. However, in almost all data presentation applications, the usefulness of a drawing of a graph depends on its readability, i.e., the capability of conveying the meaning of the diagram quickly and clearly. Readability issues are expressed by means of *aesthetics*, which can be formulated as optimization goals for the drawing algorithms. In general, the aesthetics depend on the graphic standard adopted and the particular class of graphs of interest. A fundamental and classical aesthetic is the minimization of crossings between edges. In polyline drawings it is desirable to avoid bends in edges. In grid drawings, the area of the smallest rectangle covering the drawing should be minimal. The display of symmetries is desirable independently of the graphic standard. It should be noticed that specific graphs are best drawn in ways that contradict the above aesthetics. For example, although the cube graph is planar, it is typically drawn with crossing edges, as shown in Fig. 4.

Research on graph drawing algorithms is spread over the broad spectrum of Computer Science, from VLSI to data base design. This bibliography constitutes a first attempt to encompass both theoretical and application oriented papers from disparate areas. However, we do not consider layout algorithms (such as several VLSI layout techniques) that have no impact on the problem of producing aesthetically pleasing drawings. As indicated in the title, this bibliography concentrates on *algorithms* for drawing graphs. It is written from a Computer Science viewpoint, and does not deal with other aspects of the problem of drawing graphs. Namely, we do not attempt to cover the large literature on the mathematical theory of embeddings of graphs, work on circuit and facilities layout, or psychological and philosophical issues of aesthetically pleasing drawings. However, introductory textbooks on graphs and algorithms, and samples of papers from related areas have been included for the reader's convenience.

In Section 2 we mention background reference material for graph drawing problems. Sections 3, 4, 5 and 6 consider in turn algorithms for drawing trees, planar graphs, directed graphs, and undirected graphs. Literature on systems for drawing graphs is outlined in Section 7. Papers on topics that do not fit the above classification are mentioned in Section 8. A list of significant open problems is given in Section 9. The Appendix contains tables that further

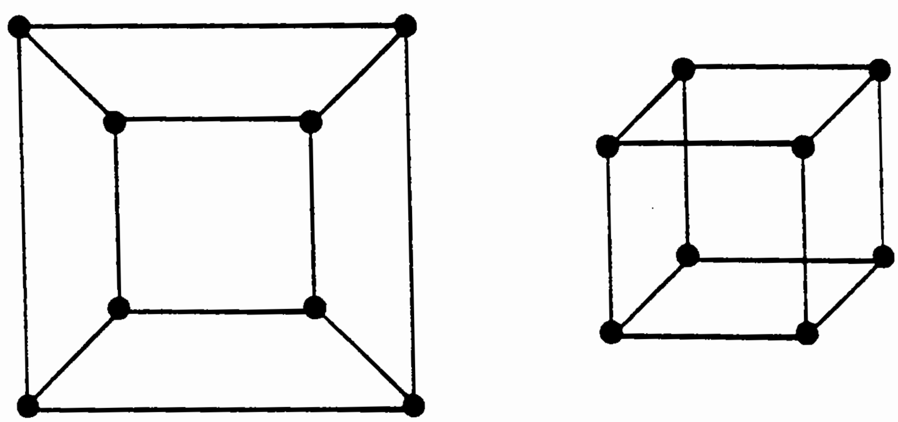

Figure 4 Two drawings of the cube graph.

classify the cited papers. Finally, an Index of the authors is provided.

Throughout the paper n and m denote the number of vertices and edges of the graph currently being considered.

2. Preliminaries

For elementary graph theory, the following textbooks may be consulted

[1] J.A. Bondy and U.S.R. Murty, *Graph Theory with Applications*, North Holland, Amsterdam, 1976.

[2] F. Harary, *Graph Theory*, Addison-Wesley, Reading, MA, 1969.

Fundamentals of data structures and algorithms are described in

[3] E. Horowitz and S. Sahni, *Fundamentals of Data Structures*, Computer Science Press, 1983.

[4] E.M. Reingold, J. Nievergelt, and N. Deo, *Combinatorial Algorithms: Theory and Practice*, Prentice Hall, 1977.

[5] S. Even, *Graph Algorithms*, Computer Science Press, Potomac, MD, 1979.

[6] R.E. Tarjan, "Data Structures and Network Algorithms," *CBMS-NSF Regional Conference Series in Applied Mathematics*, vol. 44, Society for Industrial Applied Mathematics, 1983.

Algorithms for planar graphs are presented in:

[7] T. Nishizeki and N. Chiba, *Planar Graphs: Theory and Algorithms*, Annals of Discrete Mathematics 32, North-Holland, 1988.

Concepts and applications of NP-completeness and complexity theory are described in

[8] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, 1979.

Basic concepts of computer graphics and computational geometry are given in

[9] J.D. Foley and A. van Dam, *Fundamentals of Interactive Computer Graphics*, Addison-Wesley, 1982.

[10] F.P. Preparata and M.I. Shamos, *Computational Geometry*, Springer-Verlag, New York, 1985.

3. Trees

3.1. Rooted Trees

Rooted trees are often used to represent hierarchies such as family trees, organization charts, and search trees. Planar straight-line drawings are commonly used to represent rooted trees. The following additional aesthetics are often adopted (see Fig. 5).

(1) Vertices are placed along horizontal lines according to their level (distance from the root).

(2) There is a minimum separation distance between two consecutive vertices on the same level.

(3) The width of the drawing is as small as possible.

Further, for oriented binary trees such as search trees, we require:

(4) The left and right children of each vertex v are placed to the left and right of v, respectively.

The following papers contain heuristics for drawing binary trees that address the above aesthetics. Additional aesthetics, such as centering each parent upon its children, and generating

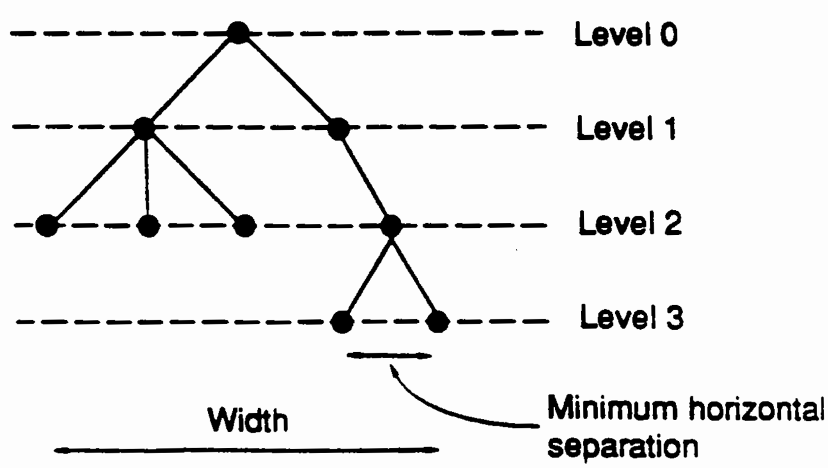

Figure 5 Drawing of a rooted tree.

congruent drawings for isomorphic subtrees, are also investigated.

[11] R.E. Sweet, "Empirical Estimates of Program Entropy," Technical Report STAN-CS-78-698, Stanford Univ., Stanford, CA, 1978.

[12] C. Wetherell and A. Shannon, "Tidy Drawing of Trees," *IEEE Trans. on Software Engineering*, vol. SE-5, no. 5, pp. 514-520, 1979.

[13] J. Vaucher, "Pretty Printing of Trees," *Software Practice and Experience*, vol. 10, no. 7, pp. 553-561, 1980.

[14] E. Reingold and J. Tilford, "Tidier Drawing of Trees," *IEEE Trans. on Software Engineering*, vol. SE-7, no. 2, pp. 223-228, 1981.

[15] J.S. Tilford, "Tree Drawing Algorithms," Technical Report UIUCDCS-R-81-1055, Dept. of Computer Science, Univ. of Illinois at Urbana-Champaign, 1981.

The extension of these algorithms to rooted trees with arbitrary vertex degree is straightforward.

The algorithms in the papers above give aesthetically acceptable drawings. However, Supowit and Reingold show that they can produce drawings much wider than necessary.

[16] K. Supowit and E. Reingold, "The Complexity of Drawing Trees Nicely," *Acta Informatica*, vol. 18, pp. 377-392, 1983.

This paper addresses the problem of constructing a minimum width drawing of a binary tree such that parents are centered upon their children and isomorphic subtrees are congruent. This problem is NP-complete if a grid drawing is required, but otherwise polynomially solvable by linear programming techniques.

A macro package for the TEX formatter that automatically produces drawings of binary trees from a textual description is presented in:

[17] A. Bruggemann-Klein and D. Wood, "Drawing Trees Nicely with TEX," Manuscript, Department of Computer Science, University of Waterloo, 1987.

The algorithm adopted is a variation of the one by Reingold and Tilford [14]. Several options are provided to specify the shape of the nodes and their labels.

An algorithm for drawing trees in a dynamic environment is presented in:

[18] S. Moen, "Drawing Dynamic Trees," Research Report LiTH-IDA-R-87-24, Dept. of Computer and Information Science, Linkoping Univ., 1987.

3.2. Free Trees

Little work has been published on drawing "free" trees, i.e., trees that do not represent hierarchies. However, the above algorithms for rooted trees can be modified to produce acceptable *radial* drawings of free trees by arranging the vertices of each level on a concentric circle about the graphtheoretic center of the tree (see Fig. 6).

Another strategy for constructing radial drawings is described in

[19] M.A. Bernard, "On the Automated Drawing of Graphs," *Prc. 3rd Caribbean Conf. on Combinatorics and Computing*, pp. 43-55, 1981.

The following paper shows how to display symmetries in radial drawings.

[20] J. Manning and M.J. Atallah, "Fast Detection and Display of Symmetry in Trees," *Congressus Numerantium*, 1989.

It is NP-complete to construct a planar orthogonal grid drawing of a tree such that the maximum edge length is minimized, as shown in:

[21] S. Bhatt and S. Cosmadakis, "The Complexity of Minimizing Wire Lengths in VLSI Layouts," *Information Processing Letters*, vol. 25, pp. 263-267, 1987.

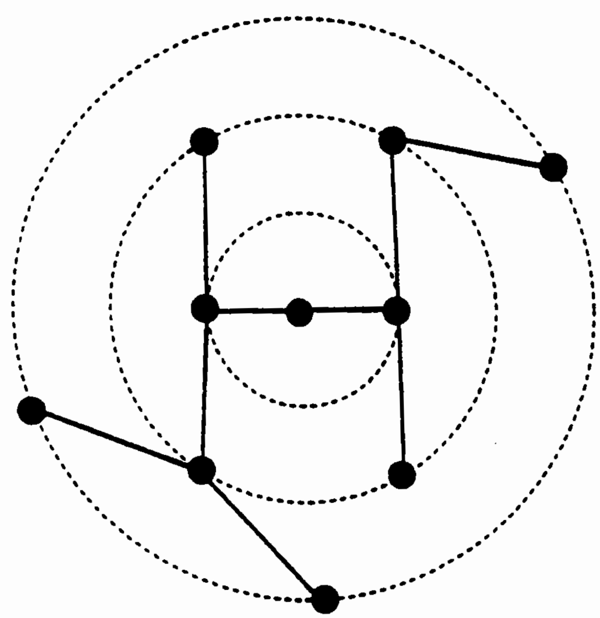

Figure 6 Radial drawing of a free tree.

[22] F.J. Bradenburg, "Nice Drawings of Graphs and Trees are Computationally Hard," Technical Report MIP-8820, Fakultat fur Mathematik und Informatik, Univ. Passau, 1988.

4. Planar Graphs

4.1. Planar Representations

A graph is *planar* if it admits a planar drawing. Planar graphs play an important role in graph theory [1,2] and graph algorithms [5]; see also:

[23] R.E. Tarjan, "Algorithm Design," *Communications ACM*, vol. 30, no. 3, pp. 205-212, 1987.

Clearly, planar drawings are aesthetically desirable. Furthermore, by adding a dummy vertex at each crossing, the problem of drawing a nonplanar graph with specified edge crossings can be reduced to drawing a planar graph.

The first algorithm to test the planarity of a graph in linear time was given in:

[24] J. Hopcroft and R.E. Tarjan, "Efficient Planarity Testing," *J. ACM*, vol. 21, no. 4, pp. 549-568, 1974.

Other approaches to testing graph planarity that yield a linear-time algorithm are presented in:

[25] A. Lempel, S. Even, and I. Cederbaum, "An Algorithm for Planarity Testing of Graphs," *Theory of Graphs, Int. Symposium*, Rome, pp. 215-232, 1966.

[26] S. Even and R.E. Tarjan, "Computing an st-Numbering," *Theoretical Computer Science*, vol. 2, pp. 339-344, 1976.

[27] K. Booth and G. Lueker, "Testing for the Consecutive Ones Property, Interval Graphs, and Graph Planarity Using PQ-Tree Algorithms," *J. of Computer and System Sciences*, vol. 13, pp. 335-379, 1976.

[28] H. de Fraysseix and P. Rosenstiehl, "A Depth-First-Search Characterization of Planarity," *Annals of Discrete Mathematics*, vol. 13, pp. 75-80, 1982.

An on-line planarity testing algorithm with logarithmic query/update time is presented in:

[29] G. Di Battista and R. Tamassia, "Incremental Planarity Testing," *Proc. 30th IEEE Symp. on Foundations of Computer Science*, pp. 436-441, 1989.

A *planar representation* is a data structure representing the combinatorial adjacencies between the faces of a planar drawing. The aforementioned planarity testing algorithms can be modified to construct a planar representation of a planar graph. The following paper extends the algorithm of [27] in this way.

[30] N. Chiba, T. Nishizeki, S. Abe, and T. Ozawa, "A Linear Algorithm for Embedding Planar Graphs Using PQ-Trees," *J. of Computer and System Sciences*, vol. 30, no. 1, pp. 54-76, 1985.

In the remainder of this Section we consider drawing algorithms that construct a planar drawing from a given planar representation. Note that a simple planar graph has no more than $3n-6$ edges.

4.2. Straight-Line Drawings

A classical result independently established by Wagner, Fary and Stein shows that every planar graph admits a planar straight-line drawing.

[31] K. Wagner, "Bemerkungen zum Vierfarbenproblem," *Jber. Deutsch. Math.-Verein*, vol. 46, pp. 26-32, 1936.

[32] I. Fary, "On Straight Lines Representation of Planar Graphs," *Acta Sci. Math. Szeged*, vol. 11, pp. 229-233, 1948.

[33] S.K. Stein, "Convex Maps," *Proc. Amer. Math. Soc.*, vol. 2, pp. 464-466, 1951.

This result also follows from Steinitz's theorem on convex polytopes in three dimensions.

[34] E. Steinitz and H. Rademacher, *Vorlesung uber die Theorie der Polyeder*, Springer, Berlin, 1934.

Convex drawings of planar graphs, i.e., planar straight-line drawings where every face is drawn as a convex polygon (see Fig. 7) were first studied by Tutte.

[35] W.T. Tutte, "Convex Representations of Graphs," *Proc. London Math Soc.*, vol. 10, pp. 304-320, 1960.

[36] W.T. Tutte, "How to Draw a Graph," *Proc. London Math Soc.*, vol. 3, no. 13, pp. 743-768, 1963.

Tutte shows that for a 3-connected graph such a drawing can be obtained by solving a system of linear equations. Thomassen characterizes the class of graphs which admit a convex drawing.

[37] C. Thomassen, "Planarity and Duality of Finite and Infinite Planar Graphs," *J. Combinatorial Theory, Series B*, vol. 29, pp. 244-271, 1980.

Chiba *et al.* show that Thomassen's result can be implemented as an algorithm for producing a convex drawing in linear time.

[38] N. Chiba, T. Yamanouchi, and T. Nishizeki, "Linear Algorithms for Convex Drawings of Planar Graphs," in *Progress in Graph Theory*, J.A. Bondy and U.S.R. Murty (Eds.), Academic Press, pp. 153-173, 1984.

[39] N. Chiba, K. Onoguchi, and T. Nishizeki, "Drawing Planar Graphs Nicely," *Acta Informatica*, vol. 22, pp. 187-201, 1985.

Becker *et al.* investigate the problem of minimizing the total edge length in a planar straight-line drawing where the external face is a prescribed convex polygon.

[40] B. Becker and G. Hotz, "On The Optimal Layout of Planar Graphs with Fixed Boundary," *SIAM J. Computing*, vol. 16, no. 5, pp. 946-972, 1987.

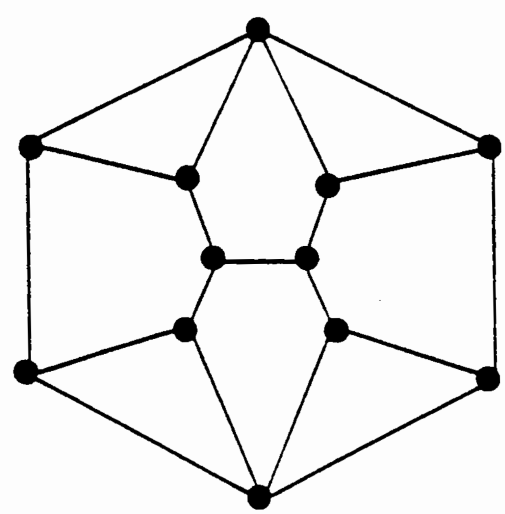

Figure 7 Convex drawing.

[41] B. Becker and H.G. Osthof, "Layout with Wires of Balanced Length," *Information and Computation*, vol. 73, pp. 45-58, 1987.

They show that the optimal drawing is unique and convex for several metrics, and can be obtained by standard numerical techniques.

An elegant linear time algorithm for constructing planar straight-line drawings has been given by Read. However, quadratic storage is required.

[42] R. Read, "New Methods for Drawing a Planar Graph Given the Cyclic Order of the Edges at Each Vertex," *Congressus Numerantium*, vol. 56, pp. 31-44, 1987.

Eades and Wormald show that the problem of constructing a planar straight-line drawing with prescribed edge lengths is NP-hard.

[43] P. Eades and N. Wormald, "Fixed Edge Length Graph Drawing is NP-hard," *Discrete Applied Mathematics*, 1989.

Manning and Atallah give an algorithm for displaying symmetries in planar straight-line drawings of planar graphs.

[44] 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 Univ., West Lafayette, IN, 1986.

[45] M.J. Atallah and J. Manning, "Fast Detection and Display of Symmetry in Embedded Planar Graphs," Manuscript, Purdue Univ., 1988.

Fraysseix *et al.* and, independently, Schnyder show that every planar graph admits a planar straight-line grid drawing with area $O(n^2)$.

[46] H. de Fraysseix, J. Pach, and R. Pollack, "Small Sets Supporting Fary Embeddings of Planar Graphs," *Proc. 20th ACM Symp. on Theory of Computing*, pp. 426-433, 1988.

[47] W. Schnyder, "Embedding Planar Graphs on the Grid," *Proc. ACM-SIAM Symp. on Discrete Algorithms*, 1990.

Chrobak shows that the constructive proof of [46] can be modified to yield an $O(n)$-time drawing algorithm.

[48] M. Chrobak, "A Linear-Time Algorithm for Drawing a Planar Graph on a Grid," Manuscript, Univ. California, Riverside, 1988.

4.3. Orthogonal Grid Drawings

Planar orthogonal grid drawings were first studied in connection with circuit layout. Within this graphic standard, minimizing the number of bends and the area is important for both diagram readability and VLSI applications (see Fig. 8).

Any planar graph of degree at most 4 admits a planar orthogonal grid drawing with area $O(n^2)$. Further, there are graphs which need quadratic area. These results are reported in:

[49] Y. Shiloach, "Arrangements of Planar Graphs on the Planar Lattice." Ph.D. Thesis, Weizmann Institute of Science, Rehovot, Israel, 1976.

[50] L. Valiant, "Universality Considerations in VLSI Circuits," *IEEE Transactions on Computers*, vol. C-30, no. 2, pp. 135-140, 1981.

Tamassia provides a $O(n^2 \log n)$ time algorithm for minimizing bends, based on network flow techniques.

[51] R. Tamassia, "On Embedding a Graph in the Grid with the Minimum Number of Bends," *SIAM J. Computing*, vol. 16, no. 3, pp. 421-444, 1987.

Storer gives three heuristics for constructing drawings with $O(n)$ bends.

[52] J.A. Storer, "On Minimal Node-Cost Planar Embeddings," *Networks*, vol. 14, pp. 181-212, 1984.

Tamassia and Tollis present another heuristic for bend minimization which has the same performance bounds as the ones by Storer and runs in $O(n)$ time.

[53] R. Tamassia and I.G. Tollis, "Efficient Embedding of Planar Graphs in Linear Time," *Proc. IEEE Int. Symp. on Circuits and Systems*, Philadelphia, pp. 495-498, 1987.

[54] R. Tamassia and I.G. Tollis, "Planar Grid Embedding in Linear Time," *IEEE Trans. on Circuits and Systems*, vol. CAS-36, no. 9, pp. 1230-1234, 1989.

NP-completeness results related to the minimization of area and total edge length in planar orthogonal grid drawings have been presented in [21, 22,52] and:

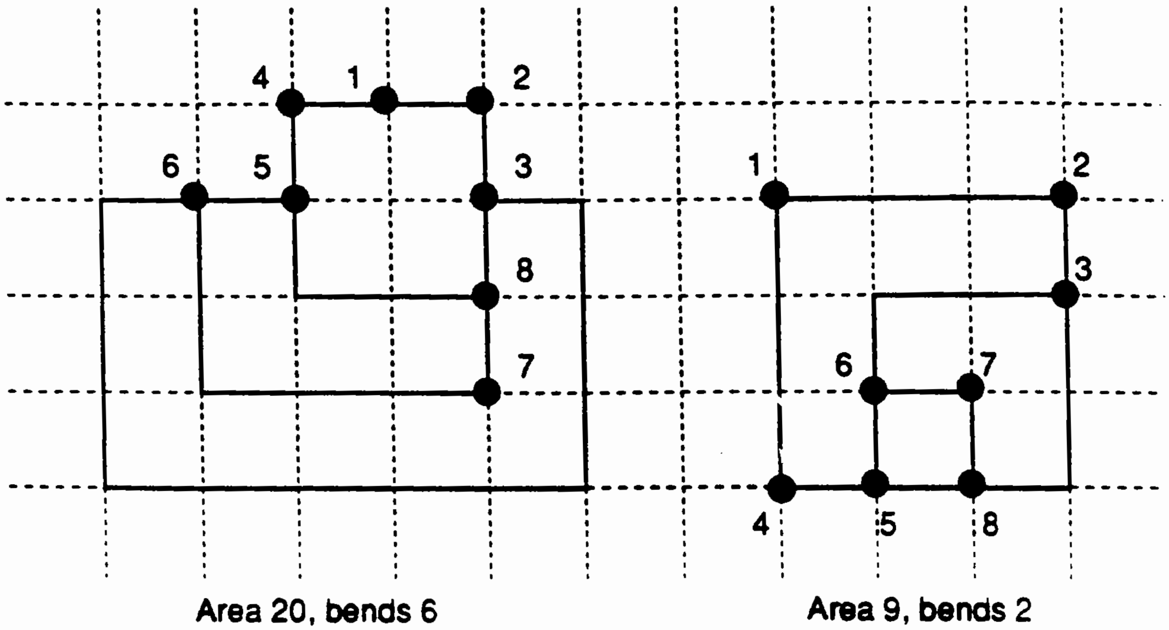

Figure 8 Examples of planar orthogonal grid drawings.

[55] 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 Inc., pp. 147-161, 1984.

This paper also gives a heuristic for area minimization.

4.4. Visibility Representations

A *visibility representation* for a planar graph G consists of representing the vertices of G by horizontal segments, and the edges of G by vertical segments, so that the edge-segment associated with each edge (u,v) intersects exactly the vertex-segments associated with u and v, and no other vertex-segment (see Fig. 9). The study of this graphic standard was originally motivated by VLSI layout and compaction problems because it gives regular and modular drawings.

[56] M. Schlag, F. Luccio, P. Maestrini, D.T. Lee, and C.K. Wong, "A Visibility Problem in VLSI Layout Compaction," in *Advances in Computing Research, vol. 2*, F.P. Preparata (Ed.), JAI Press Inc., pp. 259-282, 1985.

Theoretical results characterizing visibility representations and variations of it appear in:

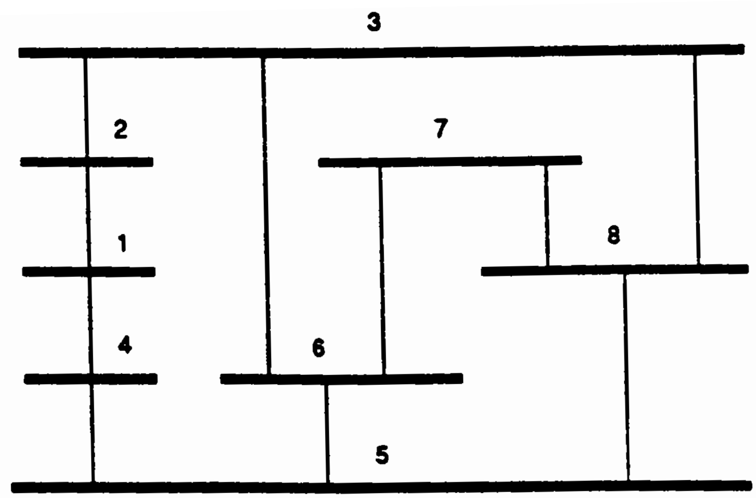

Figure 9 Visibility representation for the graph of Fig. 8.

[57] P. Duchet, Y. Hamidoune, M. Las Vergnas, and H. Meyniel, "Representing a Planar Graph by Vertical Lines Joining Different Levels," *Discrete Mathematics*, vol. 46, pp. 319-321, 1983.

[58] C. Thomassen, "Plane Representations of Graphs," in *Progress in Graph Theory*, J.A. Bondy and U.S.R. Murty (Eds.), Academic Press, pp. 43-69, 1984.

[59] S.K. Wismath, "Characterizing Bar Line-of-Sight Graphs," *Proc. ACM Symp. on Computational Geometry*, Baltimore, Maryland, pp. 147-152, 1985.

[60] R. Tamassia and I.G. Tollis, "Centipede Graphs and Visibility on a Cylinder," in *Graph-Theoretic Concepts in Computer Science*, (Proc. Int. Workshop WG '86, Bernierd, June 1986), G. Tinhofer and G. Schmidt (Eds.), Lecture Notes in Computer Science, vol. 246, Springer-Verlag, pp. 252-263, 1987.

[61] F. Luccio, S. Mazzone, and C. Wong, "A Note on Visibility Graphs," *Discrete Mathematics*, vol. 63, pp. 105-110, 1987.

[62] S.K. Wismath, "Weighted Visibility Graphs of Bars and Related Flow Problems," *Algorithms and Data Structures (Proc. WADS'89)*, pp. 325-334, Lecture Notes in Computer Science, vol. 382, Springer-Verlag, 1989.

Algorithms that construct visibility representations in linear time are given in:

[63] R.H.J.M. Otten and J.G. van Wijk, "Graph Representations in Interactive Layout Design," *Proc. IEEE Int. Symp. on Circuits and Systems*, New York, pp. 914-918, 1978.

[64] P. Rosenstiehl and R.E. Tarjan, "Rectilinear Planar Layouts of Planar Graphs and Bipolar Orientations," *Discrete & Computational Geometry*, vol. 1, no. 4, pp. 343-353, 1986.

A complete combinatorial characterization of three classes of visibility representations and linear time drawing algorithms are presented in:

[65] R. Tamassia and I.G. Tollis, "A Unified Approach to Visibility Representations of Planar Graphs," *Discrete & Computational Geometry*, vol. 1, no. 4, pp. 321-341, 1986.

4.5. Other Graphic Standards

Algorithms for constructing planar polyline grid drawings are described in [49] and:

[66] A.K. Hope, "A Planar Graph Drawing Program," *Software Practice and Experience*, vol. 1, pp. 83-91, 1971.

[67] D. Woods, "Drawing Planar Graphs," Ph.D. dissertation (Technical Report STAN-CS-82-943), Computer Science Dept., Stanford Univ., 1982.

Ozawa places vertices on a horizontal line and draws edges as half-circles or smooth connections of half-circles (see Fig. 10).

[68] T. Ozawa, "Planarity Testing for IC Layout with Constraints for Pin Order and Congestion Between Pins," *IEEE Conf. Record of the 14th Asilomar Conf. on Circuits, Systems & Computers*, pp. 188-192, 1980.

Representations of planar graphs by means of subdivisions of the plane into polygons (usually rectangles) have been studied in architectural design. Each vertex is represented by a polygon, and for each edge (u,v) the polygons associated with vertices u and v are geometrically adjacent. Essentially, this amounts to representing the graph by its dual.

[69] J.P. Steadman, *Architectural Morphology*, Pion, London, 1983.

[70] I. Rinsma, "Nonexistence of Certain Rectangular Floorplan with Specified Areas and Adjacency," *Environment and Planning B: Planning and Design*, vol. 14, pp. 163-166, 1987.

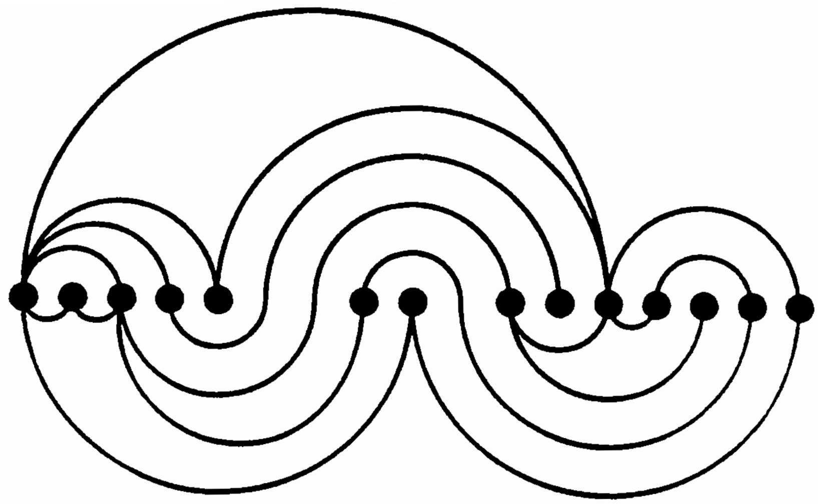

Figure 10 Planar drawing with vertices arranged on a line.

[71] J. Bhasker and S. Sahni, "A Linear Algorithm to Find a Rectangular Dual of a Planar Triangulated Graph," *Algorithmica*, vol. 3, no. 2, pp. 247-278, 1988.

In a *tessellation representation*, each constituent (vertex, edge, and face) of an embedded planar graph is represented by a rectangle with horizontal and vertical sides, and incidencies between constituents correspond to geometric adjacencies between rectangles (see Fig. 11). These representations are investigated in:

[72] R. Tamassia and I.G. Tollis, "Tessellation Representations of Planar Graphs," *Proc. 27th Annual Allerton Conf.*, 1989.

An algorithm that maps vertices to grid points to facilitate the construction of a planar drawing is described in:

[73] R. Jayakumar, K. Thulasiraman, and M.N.S Swamy, "Planar Embedding: Linear-Time Algorithms for Vertex Placement and Edge Ordering," *IEEE Trans. on Circuits and Systems*, vol. 35, no. 3, pp. 334-344, 1988.

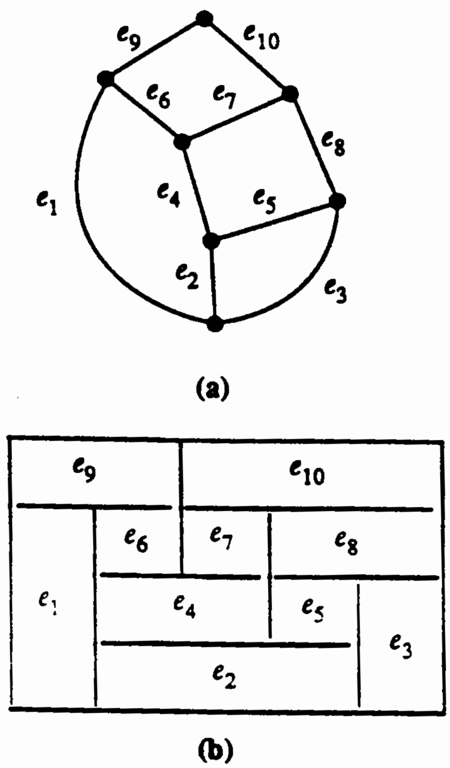

Figure 11 (a) A planar graph G. (b) Tessellation representation for G.

5. Directed Graphs

5.1. Acyclic Digraphs

Acyclic digraphs are widely used to display hierarchic structures. Examples include PERT diagrams, ISA hierarchies, and subroutine-call graphs. It is customary to represent these graphs so that the edges all flow in the same direction, e.g., from bottom to top, or from left to right (see Fig. 12). Namely, we say that a drawing of a digraph is *upward* if each arc is a curve monotonically increasing in the y-direction.

An important class of acyclic digraphs are covering digraphs of partially ordered sets. These digraphs are commonly represented by upward straight-line drawings, called *order diagrams*, *Hasse diagrams*, or simply *diagrams*.

A drawing algorithm for order diagrams is described in:

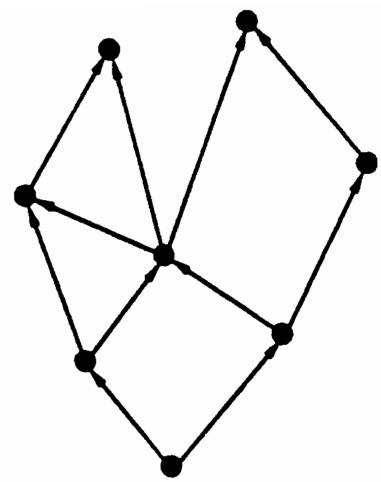

Figure 12 Upward drawing of an acyclic digraph.

[74] H. Jurgensen and J. Loewer, "Drawing Hasse Diagrams of Partially Ordered Sets," Technical Report TI 5/79, Institut fur Theoretische Informatik, Technische Hochschule Darmstad, Federal Republic of Germany, 1979.

Characterizations of planar diagrams of lattices are presented in:

[75] D. Kelly and I. Rival, "Planar Lattices," *Canadian J. Mathematics*, vol. 27, no. 3, pp. 636-665, 1975.

[76] C. Platt, "Planar Lattices and Planar Graphs," *J Combinatorial Theory, Series B*, vol. 21, pp. 30-39, 1976.

[77] D. Kelly, "Fundamentals of Planar Ordered Sets," *Discrete Mathematics*, vol. 63, pp. 197-216, 1987.

Several issues in drawing order diagrams, such as the minimization of the number of slopes used for the arcs, are investigated in:

[78] I. Rival and R. Wille, "Lattices Freely Generated by Partially Ordered Sets: Which can be Drawn?," *J. fur Reine und Angew. Math.*, vol. 310, pp. 56-80, 1979.

[79] J. Czyzowicz, A. Pelc, and I. Rival, "Drawing Orders with Few Slopes," Technical Report TR-87-12, Dept. of Computer Science, Univ. of Ottawa, 1987.

[80] J. Czyzowicz, A. Pelc, I. Rival, and J. Urrutia, "Crooked Diagrams with Few Slopes," Technical Report TR-87-26, Dept. of Computer Science, Univ. of Ottawa, 1987.

[81] A. Pelc and I. Rival, "Orders with Level Diagrams," Technical Report TR-87-11, Dept. of Computer Science, Univ. of Ottawa, March 1987.

[82] J. Czyzowicz, "Lattice Diagrams with Few Slopes," Technical Report #3, Departement D'Informatique, Univ. of Quebec at Hull, 1987.

[83] J. Czyzowicz, "Planar Lattices and the Slope Problem," Technical Report #4, Departement D'Informatique, Univ. of Quebec at Hull, 1987.

[84] J. Czyzowicz, A. Pelc, and I. Rival, "Planar Ordered Sets of Width Two," Technical Report TR-87-31, Dept. of Computer Science, Univ. of Ottawa, 1987.

Surveys on drawing techniques for order diagrams appear in:

[85] I. Rival, "The Diagram," in *Graphs and Orders*, I. Rival, Reidel Publishing, pp. 103-133, 1985.

[86] I. Rival, "Graphical Data Structures for Ordered Sets," in *Algorithms and Order*, I. Rival, Kluwer Academic Publishers, pp. 3-31, 1989.

Various combinatorial characterizations of planar straight-line upward drawings are given in:

[87] C. Thomassen, "Hasse Representations of Planar Ordered Graphs," *Order*, vol. 5, no. 4, pp. 349-361, 1989.

Efficient algorithms for constructing planar upward drawings are given in:

[88] G. Di Battista and R. Tamassia, "Algorithms for Plane Representations of Acyclic Digraphs," *Theoretical Computer Science*, vol. 61, pp. 175-198, 1988.

The display of symmetries and isomorphic subgraphs in planar upward drawings is considered in:

[89] G. Di Battista, R. Tamassia, and I.G. Tollis, "Area Requirement and Symmetry Display in Drawing Graphs," *Proc. ACM Symp. on Computational Geometry*, pp. 51-60, 1989.

This paper also shows that there exists a class of planar digraphs that require exponential area in any planar straight-line upward grid drawing.

A *hierarchical* drawing of an acyclic digraph is an upward polyline drawing where the vertices and bends are constrained to lie on a set of equally spaced horizontal lines, called *layers* (see Fig. 13). In some applications the assignment of vertices to layers is given, e.g., by the semantics of the graph. Such graphs are called *layered digraphs.*

Sugiyama *et al.* present a comprehensive approach to construct hierarchical drawings (see Fig. 14):

Step 1 Assign vertices to the layers so that vertices are distributed uniformly.

Step 2 Select a permutation of the vertices in each layer to reduce crossings.

Step 3 Adjust the position of the vertices in each layer to reduce the number of bends.

[90] K. Sugiyama, S. Tagawa, and M. Toda, "Methods for Visual Understanding of Hierarchical Systems," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. SMC-11, no. 2, pp. 109-125, 1981.

[91] K. Sugiyama and M. Toda, "Structuring Information for Understanding Complex Systems: A Basis for Decision Making," *FUJITSU Scientific and Technical Journal*, vol. 21, no. 2, pp. 144-164, 1985.

[92] K. Sugiyama, "A Cognitive Approach for Graph Drawing," *Cybernetics and Systems: An International Journal*, vol. 18, pp. 447-488, 1987.

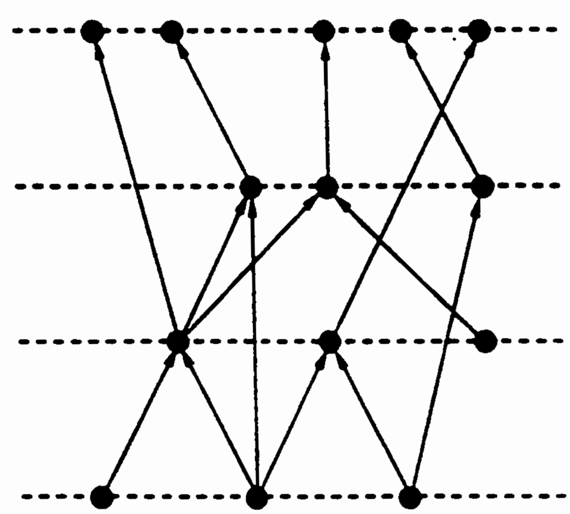

Figure 13 Hierarchical drawing.

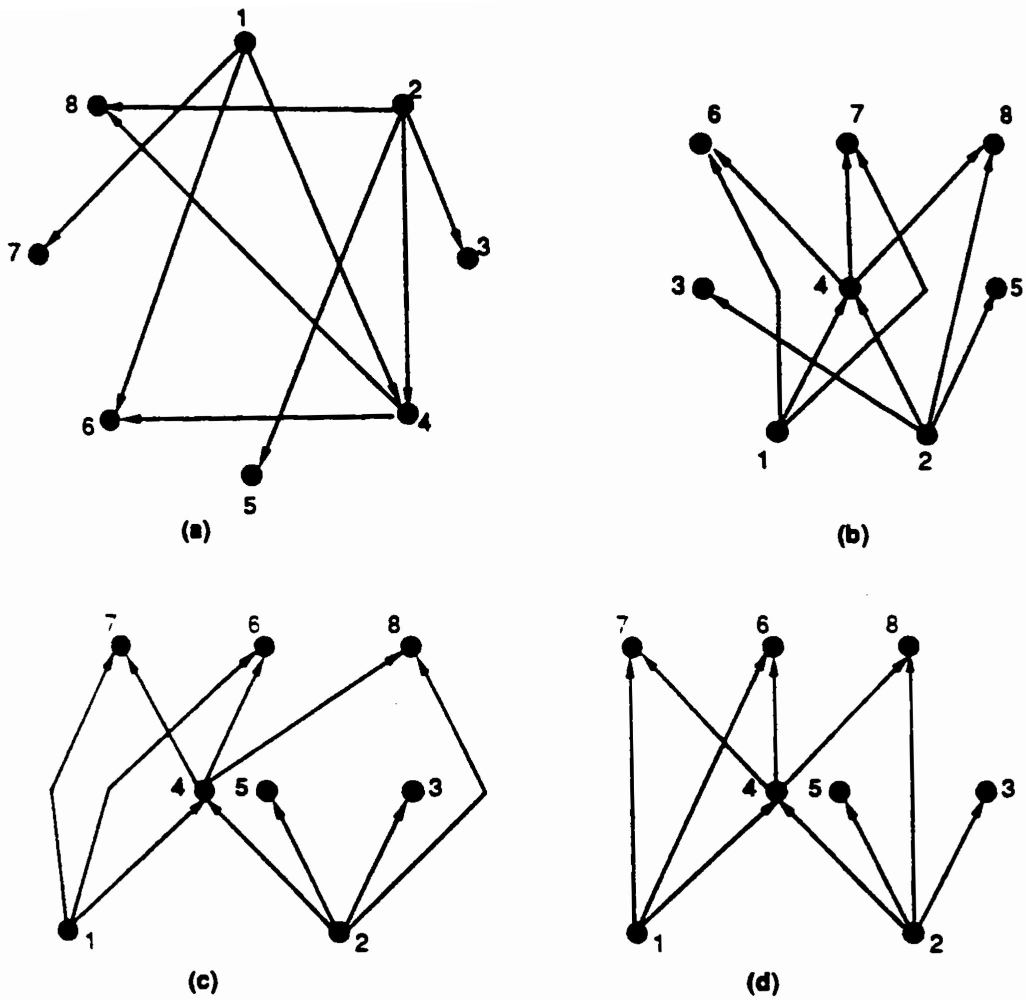

Figure 14 A general strategy for hierarchical drawings. (a) Given digraph. (b) Assignment of vertices to layers. (c) Crossing reduction. (d) Placement of vertices and bends.

Variations and extensions of this approach are presented in:

[93] M.J. Carpano, "Automatic Display of Hierarchized Graphs for Computer Aided Decision Analysis," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. SMC-10, no. 11, pp. 705-715, 1980.

[94] M.J. Carpano and M. Delarche, "Apport des Techniques Graphiques Interactives a l'Analyse Structurale de Systemes. II-Exemples de Realization et d'Application," *RAIRO Sept. Anal. Cont.*, June 1980.

[95] L.A. Rowe, M. Davis, E. Messinger, C. Meyer, C. Spirakis, and A. Tuan, "A Browser for Directed Graphs," *Software-Practice and Experience*, vol. 17, no. 1, pp. 61-76, 1987.

[96] E.B. Messinger, "Automatic Layout of Large Directed Graphs," Technical Report 88-07-08, Univ. of Washington, Dept. of Computer Science, 1988.

[97] E.R. Gansner, S.C. North, and K.P. Vo, "DAG -- A Program that Draws Directed Graphs," *Software Practice & Experience*, vol. 18, no. 11, pp. 1047-1062, 1988.

[98] D. Jablonowsky and V.A. Guarna, "GMB: A Tool for Manipulating and Animating Graph Data Structures," *Software Practice and Experience*, vol. 19, no. 3, pp. 283-301, 1989.

Heuristics for the assignment of vertices to layers in Step 1 of the above technique are described in:

[99] K. Sugiyama, "A Readability Requirement on Drawing Digraphs: Level Assignment and Edge Removal for Reducing the Total Length of Lines," Research Report No. 45, Int. Inst. for Advanced Study of Social Information Science, FUJITSU, Numazu, Japan, March 1984.

A divide-and-conquer algorithm for hierarchical drawings is proposed in:

[100] E.B. Messinger, L.A. Rowe, and R.H. Henry, "A Divide-and-Conquer Algorithm for the Automatic Layout of Large Directed Graphs," Technical Report UCB/ERL M89/23, EECS Dept., Univ. of California, Berkeley, 1989.

A recursive algorithm for hierarchical drawings that partitions the original graph into subgraphs whose elements are closely related is presented in:

[101] D.J. Gschwind and T.P. Murtagh, "A Recursive Algorithm for Drawing Hierarchical Directed Graphs," Technical Report CS-89-02, Dept. of Computer Science, Williams College, 1989.

A linear time algorithm for constructing hierarchical drawings is presented in

[102] G. Robins, "The ISI Grapher: A Portable Tool for Displaying Graphs Pictorially," Technical Report ISI/RS-87-196, Information Sciences Inst., Univ. of Southern California, 1987. (Also in Proc. Symboliikka '87, Helsinki, Finland, August 1987)

This algorithms computes independently the x- and y-coordinates of the vertices by means of simple recursive procedures.

Orthogonal hierarchical drawings are investigated in:

[103] J.E. Savage, "Heuristics for Level Graph Embeddings," *Proc. Workshop on Graphtheoretic Concepts in Computer Science*, pp. 307-318, Trauner Verlag, 1983.

Crossing reduction is a fundamental aesthetic for hierarchical drawings. An efficient algorithm to construct a planar hierarchical drawing of a layered digraph is given in:

[104] G. Di Battista and E. Nardelli, "An Algorithm for Testing Planarity of Hierarchical Graphs," in *Graph-Theoretic Concepts in Computer Science*, (Proc. Int. Workshop WG '86, Bernierd, June 1986), G. Tinhofer and G. Schmidt (Eds.), Lecture Notes in Computer Science, vol. 246, Springer-Verlag, pp. 277-289, 1987.

[105] G. Di Battista and E. Nardelli, "Hierarchies and Planarity Theory," *IEEE Trans. on Systems, Man, and Cybernetics*, 1988.

Minimizing crossings for layered digraphs is NP-hard even if there are only two layers and the positions of the vertices on the first layer are fixed, as shown in:

[106] P. Eades, B. McKay, and N. Wormald, "On an Edge Crossing Problem," *Proc. 9th Australian Computer Science Conf.*, pp. 327-334, Australian National University, 1986.

A heuristic for crossing minimization in layered digraphs with 2 layers that guarantees to produce at most three times the minimum number of crossings is given in:

[107] P. Eades and N. Wormald, "The Median Heuristic for Drawing 2-Layered Networks," Technical Report 69, Dept. of Computer Science, Univ. of Queensland, 1986.

Other heuristics for crossing minimization in layered digraphs are studied in the following papers:

[108] J. Warfield, "Crossing Theory and Hierarchy Mapping," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. SMC-7, no. 7, pp. 502-523, 1977.

[109] P. Eades and D. Kelly, "Heuristics for Reducing Crossings in 2-Layered Networks," *Ars Combinatoria*, vol. 21.A, pp. 89-98, 1986.

[110] E. Makinen, "Experiments of Drawing 2-Level Hierarchical Graphs," Technical Report A-1988-1, Department of Computer Science, Univ. of Tampere, Finland, January 1988.

[111] E. Makinen, "A Note on the Median Heuristic for Drawing Bipartite Graphs," Technical Report A-1988-4, Department of Computer Science, Univ. of Tampere, Finland, May 1988.

[112] T. Catarci, "The Assignment Heuristic for Crossing Reduction in Bipartite Graphs," *Proc. 26th Annual Allerton Conf.*, 1988.

The display of symmetries in hierarchical drawings is investigated in:

[113] K. Sugiyama, "Achieving Uniqueness Requirement in Drawing Digraphs: Optimum Code Algorithm and Hierarchic Isomorphism," Research Report No. 58. Int. Inst. for Advanced Study of Social Information Science, FUJITSU, Numazu, Japan, July 1985.

A comprehensive algorithmic framework for hierarchical drawings of digraphs is presented in:

[114] P. Eades and L. Xuemin, "How to Draw Directed Graphs," *Proc. IEEE Workshop on Visual Languages (VL'89)*, pp. 13-17, 1989.

Hierarchic drawings of *compound digraphs*, where edges represent both adjacency and inclusion relations are studied in:

[115] K. Sugiyama and K. Misue, "Visualizing Structural Information: Hierarchical Drawing of a Compound Digraph," Research Report No. 86, Int. Inst. for Advanced Study of Social Information Science, FUJITSU, Numazu, Japan, 1989.

Radial drawings of layered digraphs are investigated in [93] and:

[116] M.G. Reggiani and F.E. Marchetti, "A Proposed Method for Representing Hierarchies," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 18, no. 1, pp. 2-8, 1988.

5.2. Digraphs with Cycles

When the representation of "flow" in digraphs with cycles is an important aesthetic, one would like to maximize the number of arcs that are directed upward. This problem is equivalent to reversing the direction of a minimum number of arcs to make the digraph acyclic, and is NP-complete. Heuristics are discussed in [90,93,95,96,97,98,99,100,101,114]. After the transformation into an acyclic digraph, the techniques surveyed in the previous subsection can be applied.

If the representation of flow is not important, algorithms for drawing undirected graphs can be applied by ignoring the directions of the arcs.

6. Undirected Graphs

There are several aesthetics for drawings of general undirected graphs. The main examples are:

(1) display symmetry;

(2) avoid edge crossings;

(3) avoid bends in edges;

(4) keep edge lengths uniform;

(5) distribute vertices uniformly.

In general, the optimization problems associated with these aesthetics are NP-hard. Several complexity results are given in:

[117] D.S. Johnson, "The NP-Completeness Column: an Ongoing Guide," *J. of Algorithms*, vol. 3, no. 1, pp. 89-99, 1982.

[118] D.S. Johnson, "The NP-Completeness Column: an Ongoing Guide," *J. of Algorithms*, vol. 5, no. 2, pp. 147-160, 1984.

[119] M.R. Garey and D.S. Johnson, "Crossing Number is NP-Complete," *SIAM J. Algebraic and Discrete Methods*, vol. 4, no. 3, pp. 312-316, 1983.

[120] M.R. Kramer and J. van Leeuwen, "The Complexity of Wire-Routing and Finding Minimum Area Layouts for Arbitrary VLSI Circuits," in *Advances in Computing Research, vol. 2*, F.P. Preparata (Ed.), JAI Press, pp. 129-146, 1984.

[121] Z. Miller and J.B. Orlin, "NP-Completeness for Minimizing Maximum Edge Length in Grid Embeddings," *J. Algorithms*, vol. 6, pp. 10-16, 1985.

Besides time complexity limitations, the above aesthetics are also "competitive" in that the optimality of one often prevents the optimality of the others. Because of such difficulties, the approaches to general graph drawing are usually heuristic.

6.1. Straight-Line Drawings

A model for measuring the symmetry of a straight-line drawing of a graph is given in:

[122] R. Lipton, S. North, and J. Sandberg, "A Method for Drawing Graphs," *Proc. ACM Symp. on Computational Geometry*, pp. 153-160, 1985.

This paper also proposes an algorithm for constructing a straight-line drawing of a graph with as much symmetry as possible; however the algorithm requires the solution of the apparently intractable problem of computing the automorphism group of a graph. A completely different approach to symmetry display (which avoids computing automorphisms) is described in:

[123] P. Eades, "A Heuristic for Graph Drawing," *Congressus Numerantium*, vol. 42, pp. 149-160, 1984.

This algorithm, called *spring embedder*, is a heuristic based on a physical model. The straight-line standard is adopted. The drawing process is to simulate a mechanical system, where vertices are replaced by rings, and edges are replaced by springs. The springs attract the rings if they are too far apart, and repel them if the are too close.

Developments of the spring embedder are described in

[124] J.E. Cuny, D.A. Bayley, J.W. Hagerman, and A.A. Hough, "The Simple Simon Programming Environment: A Status Report," Technical Report 87-22, Dept. of Computer and Information Science, Univ. of Massachusetts, Amherst, MA, May 1987.

[125] T. Kamada and S. Kawai, "Automatic Display of Network Structures for Human Understanding," Technical Report 88-007, Dept. of Information Science, Univ. of Tokio, 1988.

[126] T. Kamada, "On Visualization of Abstract Objects and Relations," Ph.D. dissertation, Dept. of Information Science, Univ. of Tokio, 1988.

[127] T. Kamada and S. Kawai, "An Algorithm for Drawing General Undirected Graphs," *Information Processing Letters*, vol. 31, pp. 7-15, 1989.

[128] T. Kamada, "Symmetric Graph Drawing by a Spring Algorithm and its Applications to Radial Drawing," Manuscript, Dept. of Information Science, Univ. of Tokio, 1989.

A simple heuristic for constructing straight-line drawings which adds one vertex at a time is described in:

[129] H. Watanabe, "Heuristic Graph Displayer for G-BASE," Technical Report no. 17, Ricoh Software Research Center, Tokio, Japan, 1988.

Makinen considers straight-line drawings with vertices placed along the circumference of a circle. He shows that several related optimizations problems are NP-complete and gives a heuristic for reducing the maximum edge length.

[130] E. Makinen, "On Circular Layouts," *Int. Journal of Computer Mathematics*, vol. 24, pp. 29-37, 1988.

6.2. Planarization

The term *planarization* is used for two related problems: finding a planar subgraph with the maximum number of edges, and drawing a graph with the minimum number of crossings. Both problems are NP-hard. However, as shown in [29], a *maximal* planar subgraph can be constructed in time $O(m \log n)$.

Heuristics for finding a maximum planar subgraph are presented in:

[131] M. Marek-Sadowska, "Planarization Algorithms for Integrated Circuits Engineering," *Proc. IEEE Int. Symp. on Circuits and Systems*, pp. 919-923, 1978.

[132] N. Chiba, I. Nishioka, and I. Shirakawa, "An Algorithm of Maximal Planarization of Graphs," *Proc. IEEE Int. Symp. on Circuits and Systems*, pp. 649-652, 1979.

[133] E. Nardelli and M. Talamo, "A Fast Algorithm for Planarization of Sparse Diagrams," Technical Report R.105, IASI-CNR, Rome, 1984.

[134] T. Ozawa and H. Takahashi, "A Graph-planarization Algorithm and its Applications to Random Graphs," *Graph Theory and Algorithms, Lecture Notes in Computer Science*, vol. 108, pp. 95-107, 1981.

[135] R. Jayakumar, K. Thulasiraman, and M.N.S Swamy, "An Optimal Algorithm for Maximal Planarization of Nonplanar Graphs," *Proc. 1986 IEEE Int. Symp. on Circuits and Systems*, pp. 1237-1240, 1986.

[136] R. Jayakumar, K. Thulasiraman, and M.N.S Swamy, "O(n2) Algorithms for Graph Planarization," Technical Report CSD-88-01, Dept. Computer Science, Concordia Univ., 1988.

[137] L.R. Foulds, P.B. Gibbons, and J.W. Giffin, "Graph Theoretic Heuristics for the Facilities Layout Problem: An Experimental Comparison," *Operations Research*, 1985.

[138] P. Eades, L. Foulds, and J. Giffin, "An Efficient Heuristic for Identifying a Maximal Weight Planar Subgraph," *Combinatorial Mathematics IX, Lecture Notes in Mathematics*, vol. 952, pp. 239-251, 1982.

A heuristic for crossing minimization is given in:

[139] D. Ferrari and L. Mezzalira, "On Drawing a Graph with the Minimum Number of Crossings," Technical Report n. 69-11, Istituto di Elettrotecnica ed Elettronica, Politecnico di Milano, 1969.

The topological equivalence among nonplanar drawings of a graph is studied in:

[140] R.B. Eggleton, "Rectilinear Drawings of Graphs," *Utilitas Mathematica*, vol. 29, pp. 146-172, 1986.

There is an extensive mathematical literature on crossing numbers of graphs, see the following papers for references:

[141] H. Harborth and I. Mengersen, "Edges Without Crossings in Drawings of Complete Graphs," *J. Combinatorial Theory (B)*, vol. 17, no. 3, pp. 229-311, 1974.

[142] R.K. Guy, "Crossing Numbers of Graphs," *Graph Theory and Applications, Lecture Notes in Mathematics*, vol. 303, pp. 111-124, 1972.

6.3. Polyline Drawings

A comprehensive approach to the construction of orthogonal grid drawings, based on a number of graph algorithms, is presented in:

[143] C. Batini, E. Nardelli, M. Talamo, and R. Tamassia, "A Graphtheoretic Approach to Aesthetic Layout of Information Systems Diagrams," *Proc. 10th Int. Workshop on Graphtheoretic Concepts in Computer Science* (Berlin, June 1984), pp. 9-18, Trauner Verlag, 1984.

[144] R. Tamassia, G. Di Battista, and C. Batini, "Automatic Graph Drawing and Readability of Diagrams," *IEEE Transactions on Systems, Man and Cybernetics*, vol. SMC-18, no. 1, pp. 61-79, 1988.

Within this approach the drawing is incrementally specified in three phases (see Fig. 15): The first phase, *planarization*, determines the topology of the drawing. The second phase, *orthogonalization*, computes an orthogonal shape for the drawing. The third phase, *compaction*, produces the final drawing. This approach allows homogeneous treatment of a wide range of diagrammatic representations, aesthetics and constraints.

Another approach to the construction of orthogonal grid drawings, based on the results of [28] and on visibility representations (Section 4.4), is presented in:

[145] H. de Fraysseix and P. Rosenstiehl, "Structures Combinatoires pour des Traces Automatiques de Reseaux," *Proc. 3rd European Conf. on CAD/CAM and Computer Graphics, Paris*, pp. 332-337, Hermes, 1984.

An algorithm for constructing polyline grid drawings that allows the user to choose between a hierarchical drawing method and the orthogonal grid drawing technique of [144] is presented in.

[146] H. Trickey, "Drag: A Graph Drawing System," *Proc. Int. Conf. on Electronic Publishing*, pp. 171-182, Cambridge University Press, 1988.

Orthogonal grid drawings of graphs whose vertices have preassigned locations in the plane are investigated in:

[147] Y. Kajitani and H. Takahashi, "rectilinear Drawing of a Graph on a Plane with the Minimum Number of Segments," *Proc. 2nd Int. Catania Combinatorial Conf.*, 1989.

6.4. Application-Specific Algorithms

There are several drawing algorithms developed for specific applications, especially circuit schematics and information systems diagrams. In this framework, the semantics of the diagram and the conventions of the application area may put constraints on the drawing. For example, vertices representing "interfaces" in a Data Flow diagram are conventionally placed on the external boundary.

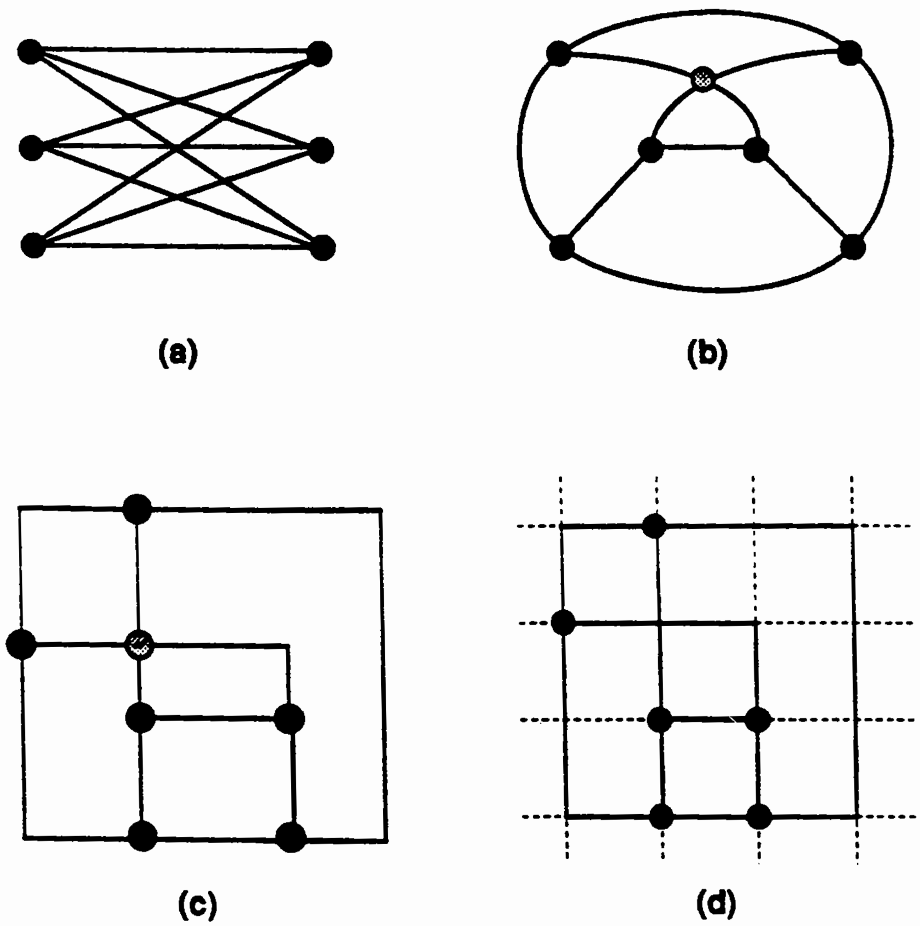

Figure 15 A general strategy for orthogonal grid drawings.
(a) Given graph. (b) Planarization. (c) Orthogonalization.
(d) Compaction.

The problem of dealing with constraints on the drawing imposed by the user is specifically investigated in:

[148] R. Tamassia, "New Layout Techniques for Entity-Relationship Diagrams," *Proc. 4th Int. Conf. on Entity-Relationship Approach*, Chicago, pp. 304-311, 1985.

The automatic generation of schematic diagrams for digital systems is studied in:

[149] A. Arya, A. Kumar, V. Swaminathan, and A. Misra, "Automatic Generation of Digital System Schematic Diagrams," *Proc. 22nd Design Automation Conf.*, pp. 388-395, 1985.

[150] F. Aoudja, M. Laborie, and A. Saint-Paul, "CASE: Automatic Generation of Electrical Diagrams," *Computer-Aided Design*, vol. 18, no. 7, pp. 356-360, 1986.

[151] M.A. Majeski, F.N. Krull, T.E. Fuhrman, and P.J. Ainslie, "Autodraft: Automatic Synthesis of Circuit Schematics," *Proc. IEEE Int. Conf. on Computer-Aided Design*, pp. 435-438, 1986.

Following the classical layout approach for integrated circuits, these algorithms perform the placement of modules and the routing of connections in two separate steps.

A drawing algorithm for PERT diagrams is presented in:

[152] G. Di Battista, E. Pietrosanti, R. Tamassia, and I.G. Tollis, "Automatic Layout of PERT Diagrams with XPERT," *Proc. IEEE Workshop on Visual Languages (VL'89)*, pp. 171-176, 1989.

An algorithms for drawing flowcharts appears in:

[153] D.E. Knuth, "Computer Drawn Flocharts," *Communications of the ACM*, vol. 6, 1963.

The following papers describe divide-and-conquer algorithms targeted toward Entity Relationship diagrams:

[154] D. Reiner, M. Brodie, G. Brown, M. Chilenskas, M. Friedell, D. Kramlich, J. Lehman, and A. Rosenthal, "A Database Design and Evaluation Workbench: Preliminary Report," *Proc. Int. Conf. on Systems Development and Requirements Specification*, Gothenburg, Sweden, 1984.

[155] D. Reiner, G. Brown, M. Friedell, J. Lehman, R. McKee, P. Rheingans, and A. Rosenthal, "A Database Designer's Workbench," in *Entity-Relationship Approach* (Proc. 5th Int. Conf. on Entity-Relationship Approach, Dijon, France, 1987), S. Spaccapietra (Ed.), North-Holland, pp. 347-360, 1987.

[156] D. Reiner and G. Brown, "Heuristic Layout for DDEW ER+ Diagrams," Manuscript, Computer Corporation of America, 1985.

Based on the general strategy of [143,144], drawing algorithms for three diagrammatic representations widely used in information systems design are given in:

[157] C. Batini, M. Talamo, and R. Tamassia, "Computer Aided Layout of Entity-Relationship Diagrams," *The Journal of Systems and Software*, vol. 4, pp. 163-173, 1984.

[158] P. Di Felice and R. Tamassia, "Automatic Layout of Flow Diagrams: Preliminary Analysis," *Proc. ISMM*, Madrid, pp. 263-267, 1985.

[159] C. Batini, E. Nardelli, and R. Tamassia, "A Layout Algorithm for Data-Flow Diagrams," *IEEE Transactions on Software Engineering*, vol. SE-12, no. 4, pp. 538-546, 1986.

7. Graph Drawing Systems

There are several computer systems available for editing graphs and graph-like diagrams. Some of these contain a naive automatic drawing facility:

[160] M. Dao, M. Habib, J. Richard, and D. Tallot, "CABRI, an Interactive System for Graph Manipulation," in *Graph-Theoretic Concepts in Computer Science*, (Proc. Int. Workshop WG '86, Bernierd, June 1986), G. Tinhofer and G. Schmidt (Eds.), Lecture Notes in Computer Science, vol. 246, Springer-Verlag, pp. 58-67, 1987.

[161] J.M Fourneau, I. Fournier, A. Germa, and D. Sotteau, "Unicom: A Computer-Aided Scratch Book for Graph Theory," Technical Report 381, L.R.I., UA410 CNRS, Univ. Paris Sud, 1987.

[162] F. Aschim and B.M. Mostue, "IFIP WG 8.1 Case Solved Using SYSDOC and SYSTEMATOR," in *Information Systems Design Methodologies: a Comparative Review* (Proc. of the IFIP WG 8.1 Working Conf. on Comparative Review of Information Systems Design Methodologies), T. Olle et al. (Eds.), North-Holland, pp. 15-40, 1982.

[163] M. Nagl and H. Zischler, "A Dialog System for the Graphical Representation of Graphs," *Applied Computer Science, vol. 13* (Proc. Workshop WG '78 on Graphtheoretic Concepts in Computer Science), pp. 325-339, 1979.

[164] M. Himsolt, "GraphEd: An Interactive Graph Editor," *Proc. STACS 89, Lecture Notes in Computer Science*, vol. 349, pp. 532-533, Springer-Verlag, 1989.

Other systems use algorithms from the above papers. They are described in [18,66,67,74,90,91,93,95,97,98,124,126,129, 144,150,151,152,154,155] and:

[165] J. Hynd and P. Eades, "The Typed Graph Editing System - TYGES," *Proc. 3rd Australasian Conf. on Computer Graphics* (*Ausgraph 85*), Brisbane, Australia, pp. 15-19, 1985.

[166] P. Eades, "SPREMB," Technical Report 85, Dept. of Computer Science, Univ. of Queensland, 1987.

[167] D. McCullough, "The Scene System," Manuscript, Dept. Computer Science, Univ. Queensland, 1989.

[168] C. Batini, E. Nardelli, M. Talamo, and R. Tamassia, "GINCOD: a Graphical Tool for Conceptual Design of Data Base Applications," in *Computer Aided Data Base Design*, A. Albano, V. De Antonellis, and A. Di Leva (Eds.), North Holland, pp. 33-51, 1985.

[169] G. Di Battista and R. Tamassia, "An Integrated Graphic System for Designing and Accessing Statistical Data Bases," *Proc. 7th Symp. on Computational Statistics (COMPSTAT 1986)*, pp. 231-236, Physica-Verlag, 1986.

[170] C. Batini, P. Brunetti, G. Di Battista, P. Naggar, E. Nardelli, G. Richelli, and R. Tamassia, "An Automatic Layout Facility and its Applications," *Proc. Int. Workshop on Software Engineering Environment*, Beijing, China, pp. 139-157, China Academic Publishers, 1986.

[171] R. Read, "Methods for Computer Display and Manipulation of Graphs, and the Corresponding Algorithms," Research Report CORR 86-12, Faculty of Mathematics, Univ. of Waterloo, July 1986.

[172] K. Nakamura, H. Fujimoto, T. Suzuki, Y. Tarui, and Y. Kiyokane, "Visual Programming Environment in Communications Software," *Proc. 5th IEEE Global Telecom Conf.*, pp. 435-439, 1986.

[173] W.F. Tichy and F.J. Newbery, "Knowledge-based Editors for Directed Graphs," in *ESEC'87 (Proc. 1st European Software Engineering Conf.)*, H.K. Nichols and D. Simpson (Eds.), Springer-Verlag, pp. 109-117, 1987.

[174] F. Newbery, "An Interface Description Language for Graph Editors," *Proc. IEEE Workshop on Visual Languages*, 1988.

[175] F. Newbery, "EDGE: An Extendible Directed Graph Editor," Technical Report 8/88, Fakultat fur Informatik, Univ. of Karlsruhe, 1988.

[176] S.P. Reiss and J.N. Pato, "Displaying Program and Data Structures," *Proc. 20th Hawaii Int. Conf. on System Sciences*, 1987.

[177] S.P. Reiss, "Integration Mechanisms in the FIELD Environment," Technical Report CS-88-18, Dept. of Computer Science, Brown Univ., 1988.

[178] S.P. Reiss, S. Meyers, and C. Duby, "Using GELO to Visualize Software Systems," Manuscript, Dept. Computer Science, Brown Univ., 1989.

[179] B. Birgisson and G. Shannon, "GraphView: A Workstation-Based Environment for Viewing Graphs and Animating Graph Algorithms," Technical Report 295, Dept. Computer Science, Indiana Univ., 1989.

8. Special Topics

Three-dimensional drawings of graphs are investigated in [98].

An optimal parallel algorithm for planarity testing is presented in:

[180] V. Ramachandran and J.H. Reif, "An Optimal Parallel Algorithm for Graph Planarity," *Proc. IEEE Symp. on Foundations of Computer Science*, 1989.

Parallel graph drawing algorithms for planar graphs are given in:

[181] R. Tamassia and J.S. Vitter, "Optimal Parallel Algorithms for Transitive Closure and Point Location in Planar Structures," *Proc. ACM Symp. on Parallel Algorithms and Architectures*, pp. 399-408, 1989.

Two notions of planarity for hypergraphs and NP-completeness results are given in:

[182] D.S. Johnson and H.O. Pollak, "Hypergraph Planarity and the Complexity of Drawing Venn Diagrams," *J. Graph Theory*, vol. 10, no. 3, pp. 309-325, 1987.

A new formalism for representing graphs and hypergraphs, called *higraph*, is introduced in:

[183] D. Harel, "On Visual Formalisms," *Communications of the ACM*, vol. 31, no. 5, pp. 514-530, 1988.

Techniques for drawing hypergraphs are presented in:

[184] E. Makinen, "How to Draw a Hypergraph," Manuscript, Dept. of Computer Science, Univ. of Tampere, 1989.

A discussion on graph drawing problems appears in:

[185] C. Esposito, "Graph Graphics: Theory and Practice," *Comput. Math. Applic.*, vol. 15, no. 4, pp. 247-253, 1988.

An experimental study of aesthetics used in Entity Relationship diagrams is reported in:

[186] C. Batini, L. Furlani, and E. Nardelli, "What is a Good Diagram? A Pragmatic Approach," *Proc. 4th Int. Conf. on the Entity Relationship Approach*, Chicago, 1985.

9. Open Problems

Despite the abundance of literature on automatic graph drawing, many theoretical and practical problems are still open. A few of the most promising directions for further research are listed below.

Performance Bounds for Planarization. Although crossing minimization is a fundamental issue, nontrivial performance bounds have not been found for any heuristic. A guaranteed heuristic would be very important both for aesthetic graph drawing and VLSI layout. Further, an algorithm with a performance guarantee for finding a maximum planar subgraph would be very welcome.

Planarity of Acyclic Digraphs. There is a combinatorial characterization of the acyclic digraphs that admit a planar upward drawing [77,88]. However, no polynomial time algorithm for testing "upward planarity" is known.

Simple Planarity Testing. The known planarity algorithms that achieve linear time complexity (Section 4.1) are all difficult to understand and implement. This is a serious limitation for their use in practical systems. A simple and efficient algorithm for testing the planarity of a graph and constructing planar representations would be a significant contribution.

General Strategy for Straight-Line Drawings. General strategies have been successfully developed for hierarchical drawings (Section 5.1) and orthogonal grid drawings (Section 6.3). These techniques take several aesthetics into account. The simplicity of straight-line drawings is very appealing, and a general straight-line drawing technique would find immediate applications.

Dynamic Drawing Algorithms. Several graph manipulation systems allow the user to interactively modify a graph by inserting and deleting vertices and edges. Data structures that allow for fast restructuring of the drawing would be very useful. Especially important is the *dynamic planarity testing* problem, where we want a data structure for planar graphs that efficiently supports the following operations:

(1) testing whether a new edge can be added while preserving planarity;

(2) adding vertices and edges which preserve planarity;

(3) removing vertices and edges.

When only insertions are allowed, this problem can be efficiently solved in logarithmic time per test or update, as shown in [29].

Appendix

We provide a set of tables that classify the cited papers/books as follows:

(1) general references;

(2) general topics in graph drawing;

(3) algorithms for constructing straight-line drawings;

(4) algorithms for constructing orthogonal drawings;

(5) algorithms for constructing polyline drawings (and drawings with other graphic standards);

(6) algorithms for planarity testing and planarization;

(7) computational complexity and lower bounds;

(8) graph theory, discrete mathematics, geometry;

(9) tools.

In each table, an asterisk in column **G**, **P**, and **U** denotes a *grid*, *planar*, and *upward* drawing, respectively.

General References					
#	Authors/Year	G	P	U	Notes
1	Bondy Murty, 76				graph theory
2	Harary, 69				graph theory
3	Horowitz Sahni, 83				algorithms
4	Reingold et al. 77				algorithms
5	Even, 79				graph algorithms
6	Tarjan, 1983				graph algorithms
7	Nishizeki Chiba, 88		*		planar graphs
8	Garey Johnson, 79				complexity theory
9	Foley van Dam, 82				computer graphics
10	Preparata Shamos, 85				computational geometry

Table 1

General Topics in Graph Drawing					
#	Authors/Year	G	P	U	Notes
85	Rival, 85			*	order diagrams
86	Rival, 89			*	order diagrams
96	Messinger, 88				includes short survey
114	Eades Xuemin, 89			*	directed graphs
144	Tamassia et al., 88				includes short survey
183	Harel, 88				visual formalisms
185	Esposito, 88				discussion on selected problems
186	Batini et al., 85				aesthetics

Table 2

Algorithms for Straight-Line Drawings					
#	Authors/Year	G	P	U	Notes
11	Sweet, 78	*	*	*	rooted trees, width
12	Wetherell Shannon,79	*	*	*	rooted trees, width
13	Vaucher, 80	*	*	*	rooted trees, width
14	Reingold Tilford, 81	*	*	*	rooted trees, width, symmetry
15	Tilford, 81	*	*	*	rooted trees, width, symmetry
16	Supowit Reingold, 83	*	*	*	rooted trees, width, symmetry
17	Bruggeman-Klein Wood, 87	*	*	*	rooted trees, width, symmetry
18	Moen, 87	*	*	*	rooted trees, width, symmetry
19	Bernard, 81		*		free trees
20	Manning Atallah, 89		*		free trees, symmetry
44	Manning Atallah, 86		*		outerplanar graphs, symmetry
45	Manning Atallah, 88		*		planar graphs, symmetry
36	Tutte, 63		*		convex faces
39	Chiba et al., 85		*		convex faces
42	Read, 87		*		simple algorithm
46	de Fraysseix et al., 88	*	*		area
47	Schnyder, 90	*	*		area
48	Chrobak, 88	*	*		area
88	Di Battista Tamassia, 88		*	*	acyclic digraphs
89	Di Battista et al., 89	*	*	*	acyclic digraphs, symmetry
93	Carpano, 80			*	acyclic digraphs, crossings
94	Carpano Delarche, 80			*	acyclic digraphs, crossings
102	Robins, 87	*		*	digraphs, area
123	Eades, 84				symmetry
124	Cuny et al., 87				symmetry
125	Kamada Kawai, 88				symmetry
126	Kamada, 88				symmetry
127	Kamada Kawai, 89				symmetry
128	Kamada, 89				symmetry
129	Watanabe, 88	*			edge length

Table 3

Algorithms for Orthogonal Drawings					
#	Authors/Year	G	P	U	Notes
49	Shiloach, 76	*	*		area
50	Valiant, 81	*	*		area
51	Tamassia, 87	*	*		bends, area
52	Storer, 84	*	*		bends, area
53	Tamassia Tollis 87	*	*		bends, area
54	Tamassia Tollis 89	*	*		bends, area
55	Dolev et al., 84	*	*		area
63	Otten van Wijk, 78	*	*		visibility representation
64	Rosenstiehl Tarjan, 86	*	*		visibility representation
65	Tamassia Tollis, 86	*	*		visibility representation
88	Di Battista Tamassia, 88	*	*	*	acyclic digraphs
103	Savage, 83	*		*	acyclic digraphs
143	Batini et al., 84	*			crossings, bends, area
144	Tamassia et al., 88	*			crossings, bends, area, constraints
145	de Fraysseix Rosenstiehl, 84	*			crossings, area
147	Kajitani Takahashi, 89	*			bends
148	Tamassia, 85	*			crossings, bends, area, constraints
149	Arya et al., 85	*			circuit schematics
150	Aoudja et al., 86	*			circuit schematics
151	Majeski et al., 86	*			circuit schematics
155	Reiner et al., 87	*			entity-relationship diagrams
156	Reiner Brown, 85	*			entity-relationship diagrams
157	Batini et al., 84	*			entity-relationship diagrams
158	Di Felice Tamassia, 85	*			flow Diagrams
159	Batini et al., 86	*			data-flow diagrams
181	Tamassia Vitter, 89	*	*		parallel algorithm

Table 4

Algorithms for Polyline Drawings & Other Graphic Standards					
#	Authors/Year	G	P	U	Notes
66	Hope, 71	*	*		simple heuristic
67	Woods, 82	*	*		area, bends
69	Steadman, 83	*	*		rectangular dual
70	Rinsma, 87	*	*		rectangular dual
71	Bhasker Sahni, 88	*	*		rectangular dual
74	Jurgensen Loewer, 79	*		*	order diagrams
88	Di Battista Tamassia, 88	*	*	*	acyclic digraphs
89	Di Battista et al., 89	*	*	*	acyclic digraphs, symmetry
90	Sugiyama et al., 81	*		*	crossings, bends, edge length
91	Sugiyama Toda, 85	*		*	crossings, bends, edge length
92	Sugiyama, 87	*		*	crossings, bends, edge length
95	Rowe et al., 87	*		*	crossings, bends, edge length
96	Messinger, 88	*		*	crossings, bends, edge length
97	Gansner et al., 88	*		*	hierarchical graphs
98	Jablonowsky Guarna, 89	*		*	crossings, bends, edge length
99	Sugiyama, 84	*		*	edge length
100	Messinger et al., 89	*		*	crossings, bends, edge length
101	Gschwind Murtagh, 89	*		*	crossings, clustering
113	Sugiyama, 85	*		*	symmetry
114	Eades Xuemin, 89	*		*	crossings, bends, edge length
115	Sugiyama Misue, 89	*		*	compound digraphs
116	Reggiani Marchetti 88			*	radial drawing, curved edges, crossings
130	Makinen, 88	*			circular drawing, edge length
146	Trickey, 88	*		*	crossings, bends, edge length
152	Di Battista et al., 89	*		*	crossings, bends, edge length
153	Knuth, 63	*	*	*	flowcharts
18[illegible]	Tamassia Vitter, 89	*	*	*	parallel algorithm

Table 5

Algorithms for Planarity Testing and Planarization

#	Authors/Year	G	P	U	Notes
23	Tarjan, 87		*		planarity testing
24	Hopcroft Tarjan, 74		*		planarity testing
25	Lempel et al., 76		*		planarity testing
26	Even Tarjan, 76		*		planarity testing
27	Booth Lueker, 76		*		planarity testing
28	de Fraysseix Rosenstiehl, 82		*		planarity testing
29	Di Battista Tamassia, 89		*		incremental planarity testing
30	Chiba et al., 85		*		construction of planar embedding
68	Ozawa, 80		*		constraints on planar embedding
73	Jayakumar et al., 88		*		construction of planar embedding
76	Platt, 76		*	*	planarity of lattices
104	Di Battista Nardelli, 87	*	*	*	planarity testing for layered digraphs
105	Di Battista Nardelli, 88	*	*	*	planarity testing for layered digraphs
107	Eades Wormald, 86			*	crossing reduction in layered digraphs
108	Warfield, 77			*	crossing reduction in layered digraphs
109	Eades Kelly, 86			*	crossing reduction in layered digraphs
110	Makinen, 88			*	crossing reduction in layered digraphs
111	Makinen, 88			*	crossing reduction in layered digraphs
112	Catarci, 88			*	crossing reduction in layered digraphs
131	Marek-Sadowska, 78		*		planarization
132	Chiba et al., 79		*		planarization
133	Nardelli Talamo, 84		*		planarization
134	Ozawa Takahashi, 81		*		planarization
135	Jayakumar et al., 86		*		planarization
136	Jayakumar et al., 88		*		planarization
137	Foulds et al., 85		*		planarization
138	Eades et al., 82		*		planarization
139	Ferrari Mezzalira, 69				crossing reduction
180	Ramachandran Reif, 89		*		parallel algorithm

Table 6

Complexity, Lower Bounds					
#	Authors/Year	G	P	U	Notes
16	Supowit Reingold, 83	*	*	*	rooted trees, width, symmetry
21	Bhatt Cosmadakis, 87	*	*		edge length
22	Bradenburg, 88	*	*		edge length
43	Eades Wormald, 89		*		prescribed edge lengths
49	Shiloach, 76	*	*		area
50	Valiant, 81	*	*		area
52	Storer, 84	*	*		area, bends, edge length
55	Dolev et al., 84	*	*		area
67	Woods, 82	*	*		area, edge length
89	Di Battista et al., 89	*	*	*	area, bends, symmetry
106	Eades et al., 86			*	crossing reduction in layered digraphs
117	Johnson, 82				various results
118	Johnson, 84				various results
119	Garey Johnson, 83				crossings
120	Kramer van Leeuwen, 84	*			area, edge routing
121	Miller Orlin, 85	*			area, edge routing

Table 7

#	Authors/Year	G	P	U	Notes
Graph Theory, Discrete Mathematics, Geometry					
31	Wagner, 36		*		planar straight-line drawings
32	Fary, 48		*		planar straight-line drawings
33	Stein, 51		*		planar straight-line drawings
34	Steinitz Rademacher, 34		*		planar straight-line drawings
35	Tutte, 60		*		convex drawings
37	Thomassen, 80		*		convex drawings
38	Chiba et al., 84		*		convex drawings
40	Becker Hotz, 87		*		minimum total edge length
41	Becker Osthof, 77		*		uniform edge lengths
56	Schlag et al., 85	*	*		visibility representations
57	Duchet et al., 83	*	*		visibility representations
58	Thomassen, 84	*	*		visibility representations
59	Wismath, 85	*	*		visibility representations
60	Tamassia Tollis, 87	*	*		visibility representations
61	Luccio et al., 87	*	*		visibility representations
62	Wismath, 89	*	*		visibility representations
64	Rosenstiehl Tarjan, 86	*	*		visibility representations
65	Tamassia Tollis, 86	*	*		visibility representations
68	Ozawa, 80	*	*		special graphic standard
72	Tamassia Tollis, 89	*	*		tessellation representations
75	Kelly Rival, 75		*	*	planarity of lattices
77	Kelly, 87		*	*	planarity of posets and acyclic diraphs
78	Rival Wille, 79			*	order diagrams, infinite digraphs
79	Czyzowicz et al., 87			*	order diagrams, few slopes
80	Czyzowicz et al., 87			*	order diagrams, few slopes
81	Pelc Rival, 87			*	order diagrams
82	Czyzowicz, 87			*	order diagrams, few slopes
83	Czyzowicz, 87		*	*	order diagrams, few slopes
84	Czyzowicz et al., 87		*	*	order diagrams, planarity
87	Thomassen, 89		*	*	planar upward drawings
122	Lipton et al., 85				symmetry
140	Eggleton, 86				theory of planarized graphs
141	Harborth, Mengersen, 74				edge crossings
142	Guy, 72				edge crossings
182	Johnson Pollak, 87		*		planarity of hypergraphs
184	Makinen, 89		*		drawings of hypergraphs

Table 8

Tools					
#	Authors/Year	G	P	U	Notes
18	Moen, 87	*	*	*	trees
66	Hope, 71	*	*		circuit schematics
67	Woods, 82	*	*		planar graphs
74	Jurgensen Loewer, 79	*		*	order diagrams
90	Sugiyama et al., 81	*		*	hierarchical graphs
91	Sugiyama Toda, 85	*		*	hierarchical graphs
93	Carpano, 80			*	hierarchical graphs
95	Rowe et al., 87	*		*	hierarchical graphs
97	Gansner et al., 88	*		*	hierarchical graphs
98	Jablonowsky Guarna, 89	*		*	hierarchical graphs
124	Cuny et al., 87				information systems
126	Kamada, 88				visualization
129	Watanabe 88	*			information systems
144	Tamassia et al., 88	*			information systems
146	Trickey, 88	*			document preparation
149	Arya et al., 85	*			circuit schematics
150	Aoudja, 86	*			circuit schematics
151	Majeski et al., 86	*			circuit schematics
152	Di Battista et al., 89	*		*	PERT diagrams
154	Reiner et al., 84				information systems
155	Reiner et al., 87				information systems
160	Dao et al., 87				graph theory
161	Fourneau et al., 87				graph theory
162	Aschim Mostue, 82				information systems
163	Nagl Zischler, 79				graph theory
164	Himsolt, 89				graph grammars
165	Hynd Eades, 85				graph theory
166	Eades, 87				computational geometry
167	McCullough, 89				graph theory
168	Batini et al., 85	*			information systems
169	Di Battista Tamassia, 86				information systems
170	Batini et al., 86				information systems
171	Read, 86				graph theory
172	Nakamura et al., 86	*		*	flowcharts
173	Tichy Newbery, 87				hierarchical graphs
174	Newbery, 88				directed graphs
175	Newbery, 88				directed graphs
176	Reiss Pato, 87				information systems
177	Reiss, 88				information systems
178	Reiss et al., 89				information systems
179	Birgisson Shannon, 89				animation

Table 9

Index of Authors