RESEARCH REPORT IIAS-RR-91-16E

Preserving the Mental Map of a Diagram

Peter Eades * Wei Lai * Kazuo Misue Kozo Sugiyama

August, 1991

International Institute for Advanced Study of Social Information Science,
FUJITSU LABORATORIES LTD.

Numazu office
140 Miyamoto, Numazu-shi, Shizuoka 410-03, Japan
Telephone: (Numazu) 0559-23-2222 Telex: 3922508J Fax: 0559-24-6180

Tokyo office
17-25, Shinkamata 1-chome, Ohta-ku, Tokyo 144, Japan
Telephone: (Tokyo) 03-3735-1111 Telex: J28448FUJITSU Fax: 03-3730-6830

Preserving the Mental Map of a Diagram

Peter Eades * Wei Lai * Kazuo Misue Kozo Sugiyama

International Institute for Advanced Study of Social Information Science,
FUJITSU LABORATORIES LTD.

140 Miyamoto, Numazu-shi, Shizuoka 410-03, Japan
Email: {misue,sugi}%iias.flab.fujitsu.co.jp@uunet.uu.net

Abstract

Many Software Engineering and Information Systems use diagrams to visualize abstract entities and relations between the entities. Recent systems have included layout algorithms, which assist a user to draw a diagram in a way that is easy to understand and remember. In an interactive system the layout often needs re-arranging on the screen, for instance, to change the central focus of the diagram. It is critical that the re-arrangement does not disturb the user's "mental map" of the drawing. In this paper we discuss several mathematical models of the user's "mental map", and present some examples of layout transformations which preserve the "mental map".

Key words: mental map, layout transformation, diagram

*Geometric Algorithms Laboratory, Department of Computer Science, University of Queensland, Queensland Australia 4072.

1 Introduction

Combinatorial models used in Software Engineering and Information Systems (such as Petri nets, conceptual schema, and dataflow machines) are often represented visually by diagrams which essentially consist of a drawing of a graph. *Automatic layout* can release the user from the time-consuming and detail-intensive chore of generating a readable diagram. Automatic layout is becoming an integral part of many systems: see [1, 2, 3, 4, 5, 6, 7, 8, 9]. A typical diagram is in Figure 1. The state of the art in algorithms for automatic graph layout is surveyed in [10]. However, the graph drawing algorithms in current use are all quite limited in their application to *interactive* visualization.

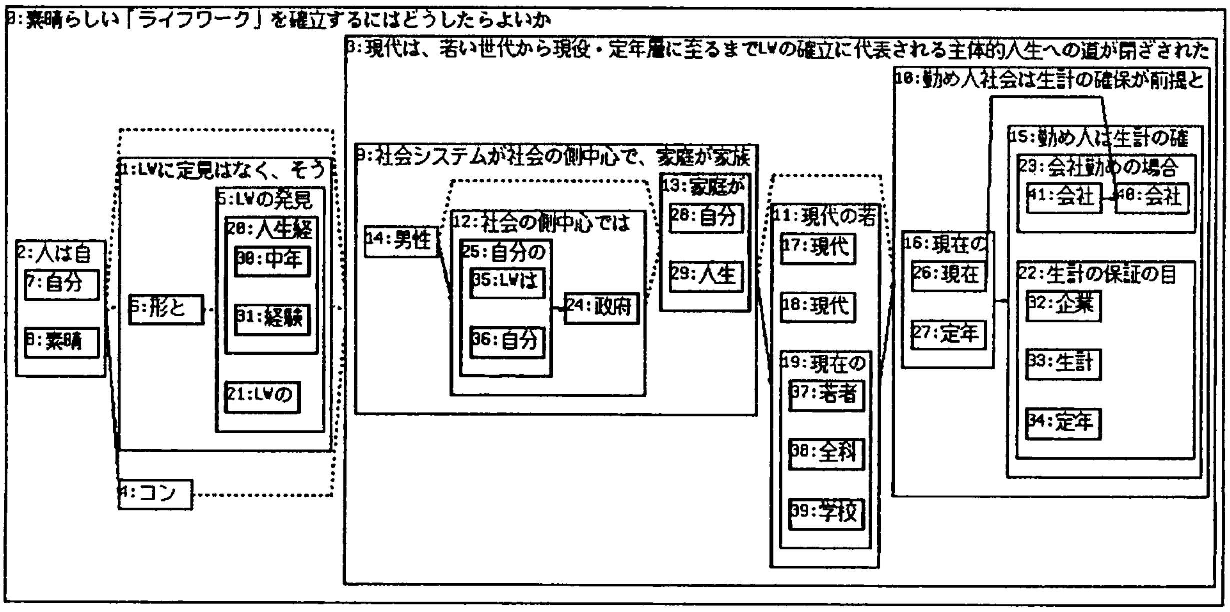

Figure 1: Sample Diagram

In an interactive visualization system, changes to the underlying model and its visual image are constantly made, sometimes by the user and sometimes by an application program. For instance, a node or a connection between nodes may be added or deleted. These changes usually spoil the layout in some local manner; for instance, an added node may overlap an existing node, or an added edge may cross existing edges. The re-application of a static layout algorithm may not help, because such an algorithm may totally re-arrange the screen, destroying the user's "mental map"of the model. An example is in Figure 2. Initially, the user sees Figure 2(a), and then an edge connecting nodes A and C is added. The re-application of a classical planarity-based layout algorithm (such as used in GIOTTO [7, 8]) will avoid the line crossing, and result in Figure 2(b). However, the re-arrangement of nodes is so severe in Figure 2(b) that the user's "mental map"of the screen is completely destroyed.

Similarly, in browsing large diagrams, a change in the point of focus is often necessary. The

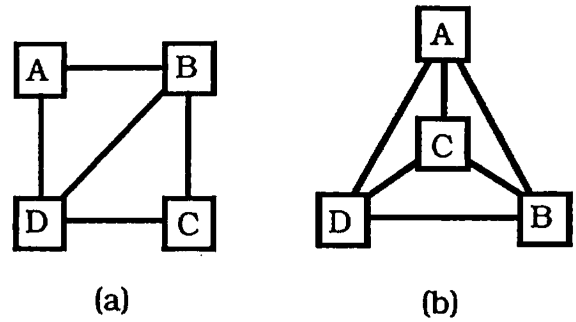

Figure 2: Adding an edge destroys Mental Map

user may become interested in a particular part of the diagram, which needs to be geometrically expanded to take a closer look at the details. Current mechanisms, such as scrolling, often hide part of the diagram and inevitably lose the user's "mental map"; we must retain the whole diagram on the screen. Thus there is a need for methods which re-arrange the screen to focus on a particular region, while not disturbing the layout so much as to lose the "mental map".

Newbery [1] refers to the problem of preserving the user's "mental map" as the *stability problem*, and presents methods which, when a change to the diagram is made, move as few nodes as possible, and move them as little as possible. This paper proposes several other models of the user's mental map, perhaps a little more global than Newbery's model. We aim at the preservation of the "shape" of the diagram. For instance, a simple scaling operation such as illustrated in Figure 3 moves *every* node in the change from Figure 3(a) to Figure 3(b); however, because "shape" is preserved, the user's "mental map" is hardly disturbed at all.

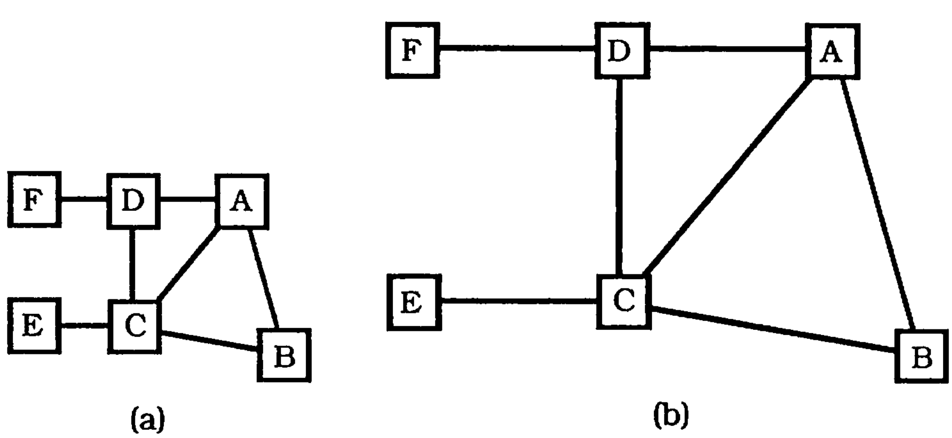

Figure 3: Uniform Scaling preserves Mental Map

The concepts of "shape" and "mental map" are intuitive. The main purpose of this paper is to suggest several possible mathematical models of the user's mental map, and thus make it possible to formally specify success or otherwise of re-arrangement algorithms for diagrams. We illustrate the viability of the models with some examples of successful re-arrangement algorithms.

This paper focuses only on *geometric* attributes of diagrams. Non-geometric attributes, such as color and linestyles, are relatively easy to preserve in transformations.

2 The Mathematical Models

In this section we will introduce several feasible mathematical models for the user's mental map of a diagram.

We will use a simplified model of a diagram. A diagram consists of a set of display objects (sometimes called "icons") on the screen. The display objects may represent the nodes of a graph. In reality, the display objects consist of various shapes, such as circles and rectangles. However, the models below are described as if the display objects are just points; in practice, these points are reference points (for example, bottom left hand corners) for actual display objects.

Each model M consists of a definition of an *equivalence relation* $\equiv_M$ on the set of finite sets S of points in the plane. A transformation t maps each point p in S to a point $t(p)$, to form a new set $T = t(S)$. Thus for a particular model M, we say that the transformation *preserves mental map under model* M, or just *preserves* M, if $S \equiv_M T$.

2.1 Orthogonal Ordering

The most basic mental map is, in Sesame Street terms, preserving up, down, left and right. More precisely, we define $S \equiv_{OO} T$ if for each pair p, q of points in S,

- $x_p < x_q$ if and only if $x_{t(p)} < x_{t(q)}$, and
- $y_p < y_q$ if and only if $y_{t(p)} < y_{t(q)}$,

where $p = (x_p, y_p)$, $q = (x_q, y_q)$, $t(p) = (x_{t(p)}, y_{t(p)})$ and $t(q) = (x_{t(q)}, y_{t(q)})$.

If $S \equiv_{OO} T$ then we say that S and T have the same *orthogonal ordering.*

Another formulation gives further intuition for its significance. Suppose that $\alpha(S)$ is an $|S| \times |S|$ matrix whose *pq*th entry is one of N, NE, E, SE, S, SW, W, NW, depending on whether the direction of q from p is North, North-East, East, South-East, South, South-West, West, or North-West. Preservation of these directional relations is clearly a very desirable property for transformations. Note that $\alpha(S) = \alpha(T)$ if and only if $S \equiv_{OO} T$; thus the Orthogonal Ordering model captures these directional relations.

2.2 Clusters

It is clear that a re-arrangement of the screen should preserve "clusters"; items which are close together should stay close together. There are many mathematical notions of *clustering* of points in the plane which could be used to make this idea precise; see [11], for example.

Given a positive real number ϵ, define an equivalence relation on S with p and q equivalent whenever $d(p,q) \leq \epsilon$. An *ϵ-cluster* of S is an equivalence class under this definition of equivalence. Thus S may be uniquely partitioned into ϵ-clusters; such a partition is an *ϵ-clustering* of S.

Suppose that for some ϵ_1 there is an ϵ_2 such that for every ϵ_1-cluster C of S, its image $t(C)$ is an ϵ_2-cluster of T, and for every ϵ_2-cluster D of T, its pre-image $t^{-1}(D)$ is an ϵ_1-cluster of S. Then we write $S \equiv_{(\epsilon_1,\epsilon_2)-CL} T$. Of course this is not very meaningful if ϵ_1 and ϵ_2 are too large (for example, if $\epsilon_1 \geq \max_{p,q\in S} d(p,q)$) or too small (for example, if $\epsilon_1 \leq \min_{p,q\in S} d(p,q)$). We suggest that

$$\epsilon_1 = \max_{p\in S} \min_{q\in S, q\neq p} d(p,q)$$

is suitable.

More sophisticated notions of clustering are perhaps better. For instance, several notions of the "shape" of a set of points are discussed in [12]. The general idea is to define a graph G whose nodes are the points of S, and whose edges are defined by metric relations between the points. Simple concepts of the "shape" of a set of points include the *convex hull* and the *minimum spanning tree* of the set. A more interesting concept is the *sphere of influence graph*, which has an edge between p and q if and only if

$$d(p,q) \leq \min_{r\in S} d(p,r) + \min_{s\in S} d(q,s).$$

If S and T have the same sphere of influence graph, then we write $S \equiv_{SOI} T$. Toussaint [13] argues that the sphere of influence graph accurately captures the "low level perceptual structure of visual scenes of dot patterns".

2.3 Topology

A transformation is a *homeomorphism* if it preserves "open sets" in the topological sense (see [14]). An important property of a homeomorphism is that it maps the inside of a closed continuous curve to the inside of a closed continuous curve: intuitively, a homeomorphism preserves "insideness". We write $S \equiv_T T$ if there is a homeomorphism which maps S to T.

Some specific homeomorphisms are particularly interesting. A linear function from the plane to itself is a *congruence* if it is either a reflection, a rotation, a translation, a scaling, or a combination of these. We say that S and T are *congruent* and write $S \equiv_C T$ if there is a congruence which maps S to T. Intuitively, $S \equiv_C T$ if we can shrink or expand S, pick it up, and lay it on top of T. Thus congruence is a very strong model for mental map. However, the difficulty of achieving congruence makes it perhaps impractical.

3 Some Sample Transformations

In this section we illustrate some transformations, and consider under which models they preserve mental map.

These sample transformations are oriented toward two interactive layout problems.

The *overlap problem* is to eliminate overlapping of display objects (as in Figure 4). Overlapping often occurs when adding or moving display objects. We need a re-arrangement of the screen which removes the overlaps yet does not disturb the mental map.

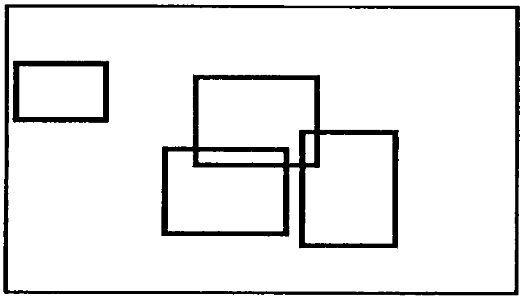

Figure 4: Diagram with some overlaps among display objects

The *whole and detailed view problem* arises when we want to focus attention on a particular region of a large diagram. We must produce a layout which gives a detailed view of region of interest, while retaining a view of the whole diagram in less detail. Most visualization systems have difficulty in obtaining both these aspects when viewing large diagrams (such as Figure 5) on an ordinary screen. Scrolling is not an adequate mechanism because (as in Figure 5) it hides some of the diagram.

3.1 Scaling

Uniform scaling may be used to eliminate intersections among display objects. We choose a center (a, b) and a scale factor s, and map the position (x, y) of every display object to $(a + s(x - a), b + s(y - b))$; the size and the shape of display object are not changed. By choosing

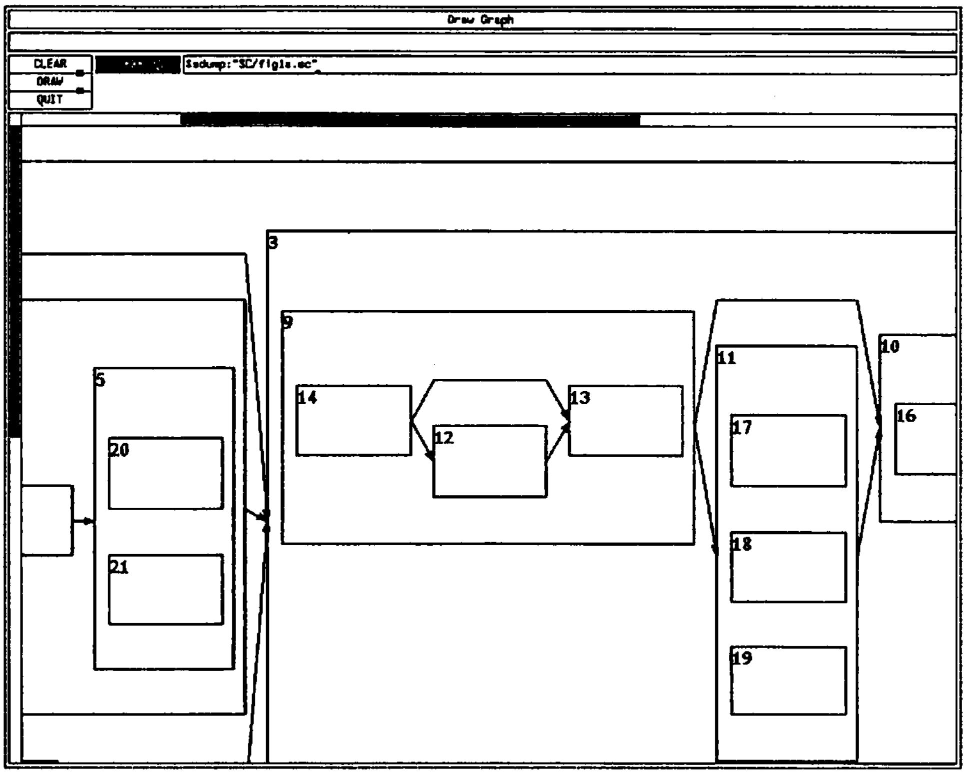

Figure 5: A diagram which is too large for the screen

s large enough, all overlaps between display objects can be removed. Uniform scaling preserves mental map under each of the models in the previous section. However, it is not useful in practice because it usually makes the diagram too large for the screen. (See Figure 6.)

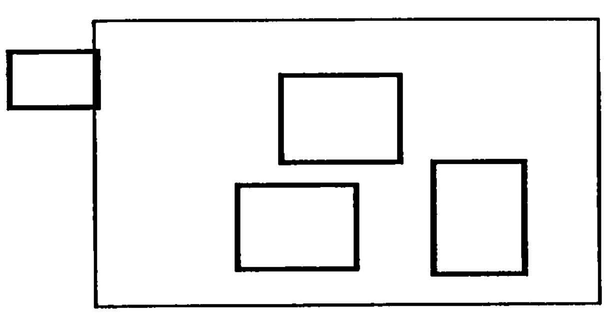

Figure 6: Diagram transformed by the uniform scaling

The *biform display method* [15] is a more useful form of scaling. It is primarily for the whole and detail views problem. In this method, *viewpoint areas* can be chosen to be shown in detail. The algorithm magnifies uniformly in each viewpoint area, and reduces uniformly outside the viewpoint areas, while preserving the outer rectangular frames of the diagram (see Figure 7). The magnification ratios for each axis are equal so that viewpoint areas appear on the same scale.

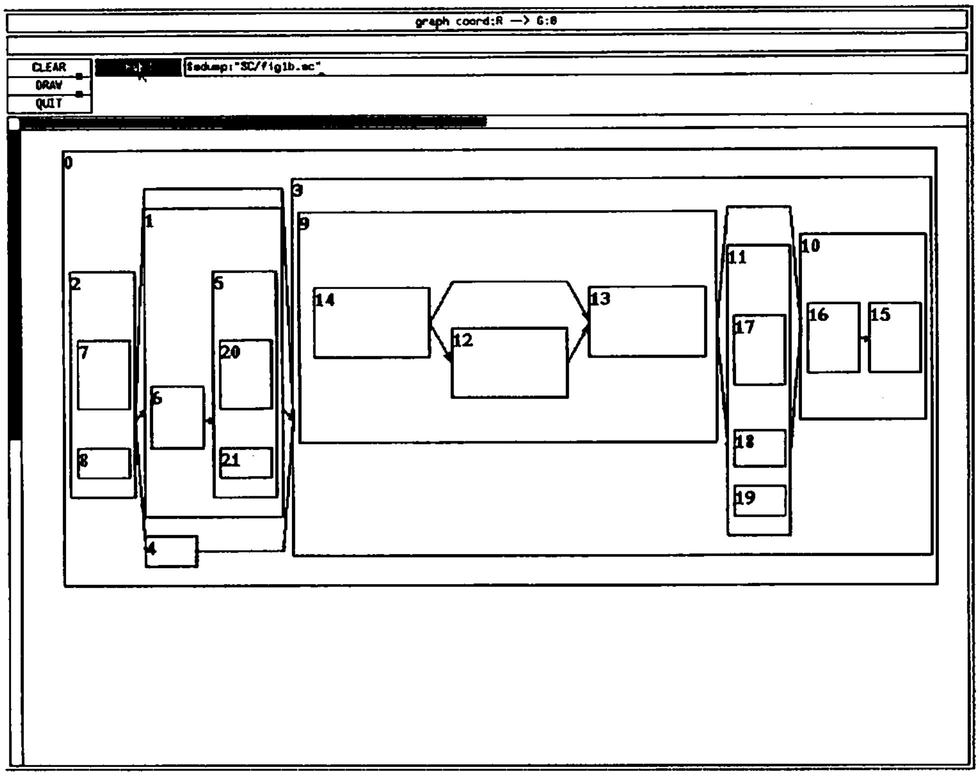

Figure 7: Representation in the biform display method

The biform display method preserves topology and orthogonal ordering.

3.2 Horizontal Shuffle and Force-scan

The *horizontal shuffle* may be used to eliminate overlaps between the display objects. In the horizontal shuffle, we scan the left hand edges of the the enclosing rectangles of display objects from left to right. If we come to an enclosing rectangle R which intersects other enclosing rectangles, then we move all display objects whose left hand edges are to the right of the left hand edge of R. We make the movement only in the x direction, and only far enough to avoid overlaps with R.

The horizontal shuffle preserves orthogonal ordering, while keeping the diagram within a small area. (See Figure 8.)

The *force-scan* algorithm [16] is an extension of the horizontal shuffle. Effectively, the algorithm shuffles the rectangles in both directions using an algorithm similar to the horizontal shuffle as a basic step. Intuitively, rectangles exert a "force" on each other which pushes them apart, or pulls them together.

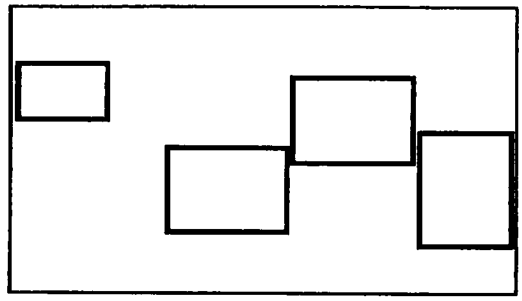

Figure 8: Diagram transformed by the horizontal shuffle

The force-scan algorithm also preserves orthogonal ordering, and it performs better than the Horizontal Shuffle with respect to area. (See Figure 9.)

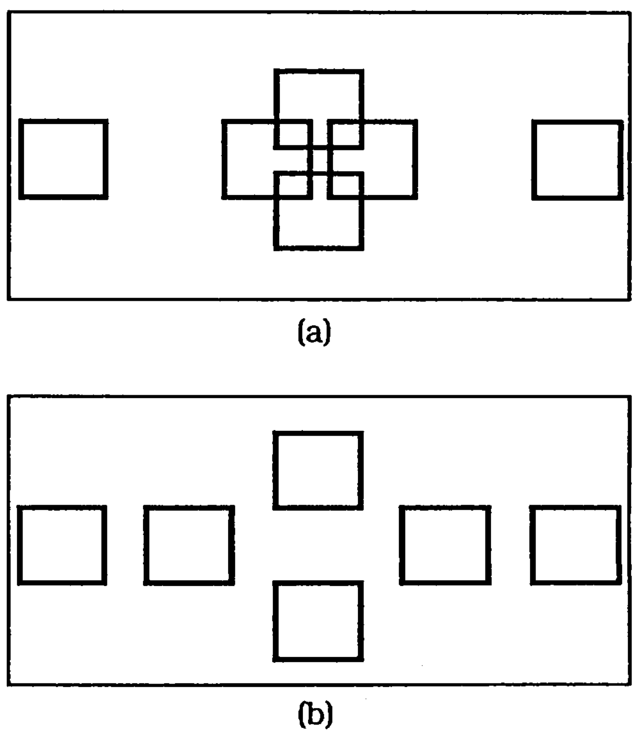

Figure 9: Diagram transformed by the force-scan

4 Conclusion

We have presented a few mathematical models of the user's "mental map". These models are useful for determining the performance of tools for interactive diagram layout. We have shown that there are algorithms which preserve orthogonal ordering and topology. we would like to see algorithms which preserve clustering as well.

References

[1] F. Newbery Paulich and W. F. Tichy. Edge: An extendible directed graph editor. *Software - Practice and Experience*, 20(S1):63–88, 1990.

[2] T. Kamada. *Visualizing Abstract Objects and Relations*, volume 5 of *Computer Science*. World Scientific, Singapore, New Jersey, London, Hong Kong, 1989.

[3] E. R. Gansner, S. C. North, and K. P. Vo. Dag - a program that draws directed graphs. *Software Practice and Experience*, 18(11):1047–1062, 1988.

[4] K. Misue and K. Sugiyama. On automatic drawing of compound graphs for computer aided diagrammatical thinking. *Transactions of the Information Processing Society of Japan*, 30(10):1324–1334, 1989. (In Japanese).

[5] K. Sugiyama and K. Misue. Visualization of structural information: Automatic drawing of compound graphs. *IEEE Transactions on Systems, Man and Cybernetics*, 13(2), 1991. (to appear).

[6] L. A. Rowe, M. Davis, E. Messinger, C. Meyer, C. Spirakis, and A. Tuan. A browser for directed graphs. *Software Practice and Experience*, 17(1):61–76, 1987.

[7] C. Batini, G. Di Battista, and R. Tamassia. Automatic graph drawing and readability of diagrams. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-18(1):61–79, 88.

[8] C. Batini, M. Talamo, and R. Tamassia. Computer aided layout of entity relationship diagrams. *Journal of Systems and Software*, 4:163–173, 84.

[9] H. Trickey. Drag: A graph drawing system. In *Proceedings of the International Conference on Publishing Systems*, pages 171–182. Cambridge University Press, 1988.

[10] P. Eades and R. Tamassia. Algorithms for drawing graphs: an annotated bibliography. Technical Report CS-89-09, Department of Computer Science, Brown University, 1989. to appear in *Networks*.

[11] J. T. Tou and R. C. Gonzalez. *Pattern Recognition Principles*. Addison Wesley, 1974.

[12] G. T. Toussaint. *Computational Morphology*. North Holland, 1988.

[13] G. T. Toussaint. A graph-theoretic primal sketch. In *Computational Morphology*, pages 229–259. North Holland, 1988.

[14] N. Dunford and J. T. Schwartz. *Linear Operators.* Interscience (Wiley), 1957.

[15] Kazuo Misue and Kozo Sugiyama. Multi-viewpoint perspective display methods: Formulation and application to compound graphs. In *Proceedings of HCI International 1991*, 1991.

[16] P. Eades and Wei Lai. Automatic display of network structures with disjoint node images. manuscript, 1991.