Higher Dimensional Representations of Graphs

Andreas Buja and Nathaniel Dean
AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974

Michael L. Littman
Dept. of Computer Science
Brown University
Providence, RI 02912

Deborah Swayne
Bellcore
445 South Street
Morristown, NJ 07960-1910

May 31, 1995

Abstract

Graphs are often used to model complex systems and to visualize relationships, and this often involves drawing a graph in the plane. For this, a variety of algorithms and mathematical tools have been used with varying success. We demonstrate why it is often more natural and more meaningful to view higher dimensional representations of graphs. We present some of the theory and problems associated with constructing such representations, and we briefly describe some visualization tools which are now available for experimental research in this area.

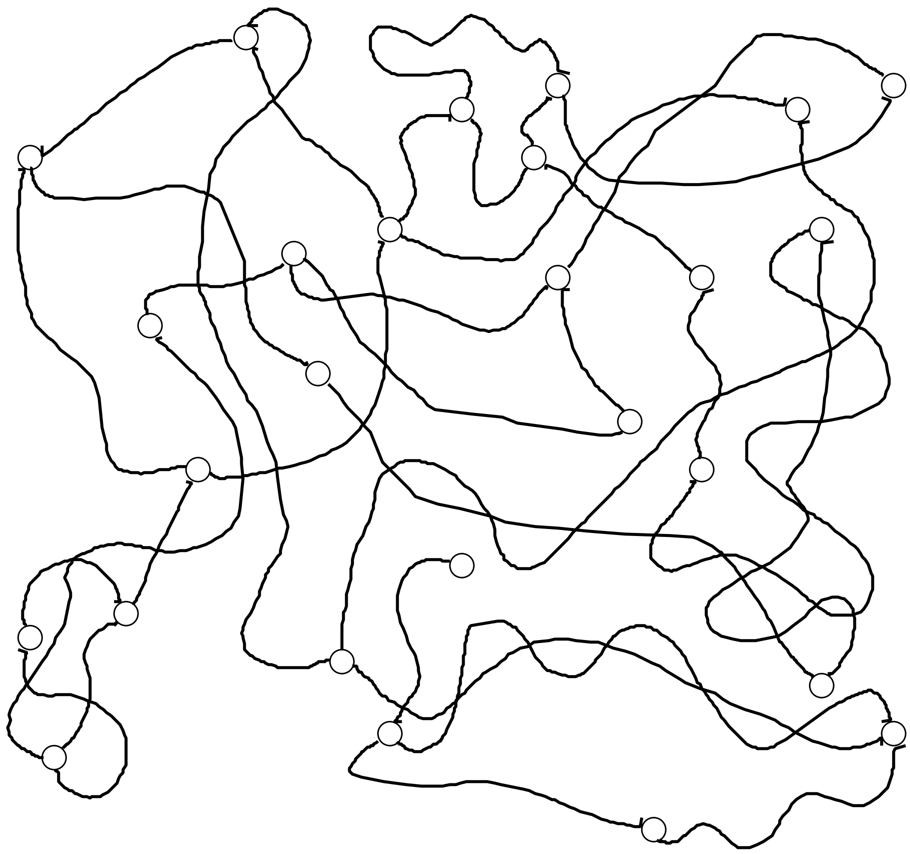

Figure 1: Picture of a graph

1 Introduction

For many people the right picture is the key to understanding. The various ways of visualizing a graph provide different insights, and often hidden relationships are revealed. Also, the representation of a graph enfluences how well it can be used. We focus on several mathematical problems associated with drawing graphs. The problems and methods of solution are as diverse as the objectives including constraints on the area of the drawing, the length and shape of edges, and beauty.

What is a graph? We quote one of the standard texts [2] directly:

> A *graph* G is an ordered triple $(V(G), E(G), \psi_G)$ consisting of a nonempty set $V(G)$ of *vertices*, a set $E(G)$, disjoint from $V(G)$, of *edges*, and an *incidence function* ψ_G that associates with each edge of G an unordered pair of (not necessarily distinct) vertices of G. If e is an edge and u and v are vertices such that $\psi_G(e) = uv$, then e is said to *join* u and v; the vertices u and v are called the *ends* of e.

Actually, this is really not what we imagine when we speak of graphs. We usually imagine points, dots or circles drawn in the plane to represent the vertices of the graph and curves joining various pairs of vertices to represent the edges (see Figure 1). Our particular picture of a given graph may vary depending upon various factors including the problem under consideration and any known properties of the graph. If we actually take the time to draw the graph on paper, then we would probably take into account various qualities of the drawing that might enhance our understanding of it structure, make it more appealing, or somehow enhance problem solving. The drawing of the graph in Figure 1, for example, might be improved by straightening some of the curved edges, redrawing to minimize crossing, positioning the vertices more symmetrically, and so on. However, the quality of the drawing really depends on factors that vary with the application. For example, consider the three drawing of the 5x5 grid graph shown in Figure 2. If our objective was to maximixe the number of crossings, then the first drawing would clearly be the winner.

Much theory is associated with constructing different types of representations, and some of the highlights are discussed in this paper. The theory primarily focusses on 2-dimensional representations, but we demonstrate how the same ideas can be applied in higher dimensions. For example, Sections 2.2, 2.3 and 5 discuss three different types of spring embeddings.

There are many applications where graphs are used to model complex systems and to visualize relationships. Scientists have largely been limited to the use of hand drawings on 2-dimensional surfaces, but with the availability of sophisticated technologies it is now feasible to generate drawings with a computer and to view graphs in higher dimensions. Of course, the problem of representation still remains. Each application or theoretical problem imposes its own set of objectives or constraints on the layout or what would make a desirable drawing. In general, each representation offers new insight into the structure of the graph. Any of them may be useful in situations where others may not. We've also learned that viewing and exploring graphs embedded in a higher dimensional space is a nontrivial task and probably requires some practice or training. To view and experiment with various representations we have developed some computer tools which so far have proved to be quite helpful. We briefly describe them in Section 6.

2 Problems in Distance Geometry

In this section we present several problems in distance geometry associated with drawing graphs in higher dimensional spaces. Idealized mathematical abstractions such as these help to realize that there are some limitations on what we can accomplish algorithmically.

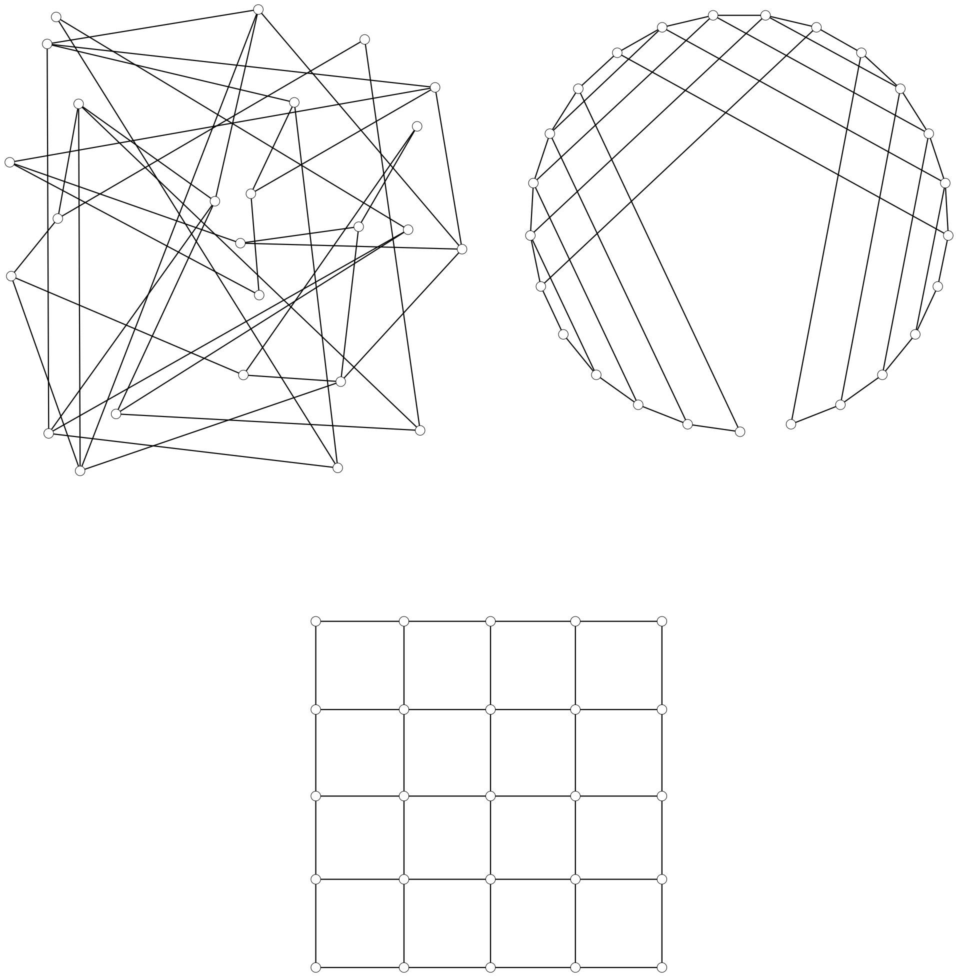

Figure 2: “Nice” Drawings

2.1 Unit Distance Graphs

A **unit-distance graph** in $\Re^d$ is a graph whose vertices can be assigned distinct positions in $\Re^d$ so that two vertices are adjacent if and only if their Euclidean distance is 1. Chilakamarri[7] has characterized the unit-distance graphs in $\Re^d$ in terms of Z^d. The main open problem in this area concerns coloring unit-distance graphs; that is, we assign a color to each vertex of the graph so that no two adjacent vertices get the same color. The chromatic number $\chi(G)$ of G is the smallest integer k such that G is k-colorable. Using a tesselation of the plane with regular hexagons of diameter 1 and a 7-coloring of the hexagons where each point in a hexagon gets the same color, it is easy to see that $\chi(G) \leq 7$ for any unit-distance graph G (see [24]).

Open Problem 1 *Find the best possible upper bound for the chromatic number of unit-distance graphs in $\Re^d$ for each integer $d \geq 2$.*

A related notion is the dimension of a graph [16]. The **dimension** of a graph G is the smallest integer d such that G can be embedded in the d-dimensional euclidean space $\Re^d$ in such a way that every edge has length 1. In [16] the dimensions or various graphs are determined; for example, the dimensions of the complete graph K_n, the n-cube Q_n for $n \geq 2$ and the Petersen graph are $n-1$, 2 and 2, respectively. The wheel $W_{1,6}$ has dimension 2, but $\dim(W_{1,n}) = 3$ for all other $n \geq 3$.

Theorem 1 ([16])

$$\dim(K_{m,n}) = \begin{cases} 2 & \textit{if } m = 1 \textit{ and } n \geq 3 \\ 3 & \textit{if } m = 2 \textit{ and } n \geq 3 \\ 4 & \textit{if } m \geq 3 \textit{ and } n \geq 3 \end{cases}$$

Case $n = 2$ of the following problem is trivial, but the case $n = 3$ has been open now for almost 3 decades.

Open Problem 2 *[16] Characterize the graphs G of dimension n such that $\dim(G - e) < n$ for every edge e of G.*

2.2 Tutte Embedding

Evidently, the first serious investigation of graph layouts in higher dimensions was initiated by Steinitz[26]. He proved that *a graph is planar and 3-connected if and only if it is the one-skeleton of a convex 3-dimensional polyhedron.* The basic importance of this theorem to graph theorists stems from the fact that it transforms combinatorial problems about a 3-dimensional object into 2-dimensional, presumably less complicated considerations. On the other hand, this also implies that 3-connected planar graphs have a very natural geometric representation (for example, see Figure 8).

Tutte[27] proved that *given a 3-connected planar graph G with facial cycle $C = v_1v_2 \ldots v_k$ and a convex polygon with vertices $p_1, p_2, \ldots, p_k$, there exists a rectilinear* *embedding of G where v_i is located at p_i $(i = 1, \ldots, k)$ and every face boundary is a convex polygon.*

This theorem has been generalized in a couple of ways. It was generalized in [1] to prove that *although the crossing number and rectilinear crossing number of graphs are (in general) unrelated, for graphs of bounded degree, they are bounded as functions of each other.* This is yet another example of how planar graphs can be used to draw nonplanar graphs. The generalization of Tutte's theorem to higher dimensions seems suprisingly tricky. Some results are presented in [8] where some nice drawings of cell complexes such as the one shown in Figure 3 are constructed.

2.3 Connectivity

Let G be a graph with n vertices. Then G is k-connected if G is a complete graph K_n with $n \geq k+1$ or every vertex cut of G contains at least k vertices. This is the usual definition of k-connectedness, but recently a new characterization has appeared based on geometric ideas. It characterizes k-connected graphs in terms of embeddings of graphs in $\Re^{k-1}$. This result also has connections to physics since the embedding can be found by fixing the k vertices appropriately in $\Re^{k-1}$, letting the edges act like ideal springs, and letting the remaining vertices adjust their positions to settle in equilibrium.

Theorem 2 *A graph G is k-connected if and only if, specifying any k vertices of G, the vertices of G can be represented by points of $\Re^{k-1}$ so that no k are on a hyperplane and each vertex is in the convex hull of its neighbors, except for the k specified vertices.*

A similar theorem was proved in [6] for directed graphs. In this case only the out-neighbors of a vertex determine its position, not the in-neighbors.

Theorem 3 *A directed graph G is k-connected if and only if, specifying any k vertices of G, the vertices of G can be represented by points of $\Re^{k-1}$ so that no k are on a hyperplane and each vertex is in the convex hull of its out-neighbors, except for the k specified vertices.*

Both of these approaches generalize the notion of st-numbering. An st-numbering for an edge st in a graph G is a linear ordering of the vertices so that every vertex other than s and t has a neighbor to its left and one to its right. In fact, for undirected graphs with $k = 2$ the embedding is equivalent to an st-numbering. Such orderings have been used extensively in applications such as planarity testing [21] and fault-tolerant protocols for distributed computers [18]. The impact of these generalizations of st-numberings is still to be determined.

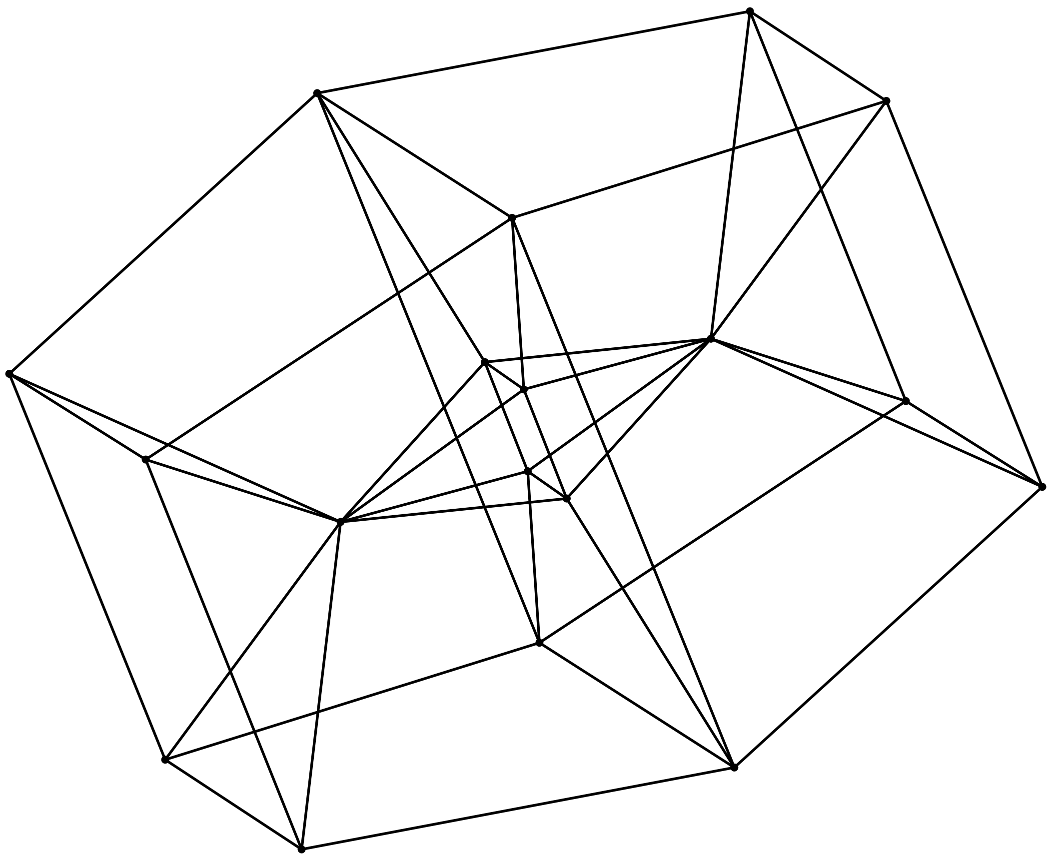

Figure 3: 3-Dimensional Tutte Embedding

3 Symmetry

For beauty, perhaps symmetry is the most important factor. This idea was investigated in [23] by constructing drawings in the plane that display as many symmetries of the graph as possible. The group of symmetries of a drawing D of a graph G is generated by the permutations associated with reflections and rotations. It is isomorphic to a subgroup of the automorphism group of G. The objective is to maximize the order of the group of symmetries. In extending this idea to three or more dimensions, of course, in general we have a better chance of approximating the automorphism group. To our knowledge no work has been done in this direction.

4 Grid Embeddings

Recently, much attention has been focused on embedding planar graphs on a small grid (see [9], [17], [20], [25]). This problem is related to certain problems in VLSI layout design. Of course, for a rectilinear drawing of any graph (even nonplanar graphs) it suffices to specify the positions of the vertices. The precision required to represent a drawing in this way is directly related to both the area of the layout and the size of the grid (or integer lattice) needed to provide a grid point for each vertex of the drawing (see Figure 4). The linear time algorithm in [25] constructs an embedding of any planar graph on the $n-2$ by $n-2$ grid. A small grid also tends to keep vertices from clustering together in a relatively small portion of the total drawing area.

One drawback of the methods used by most of the grid embedding algorithms is the fact that they first triangulate the graph, construct a grid embedding of the triangulation, and then remove the added edges. Consequently, some rather jagged layouts are produced for some generally well-behaved graphs (see Figure 5). Is there a way to avoid this? Sometimes. Chrobak [9] has shown that *every 3-connected planar graph has a convex embedding on a polynomial-size grid.* This leaves open the problem of embedding graphs efficiently in higher dimensions. In fact, so far no one has even proposed a definition of convexity for graphs embedded in 3-space. Some minor results can be found in [8].

Open Problem 3 *How can "convex" embeddings of a graph in $\Re^d$ be realized on a polynomial-size d-dimensional grid?*

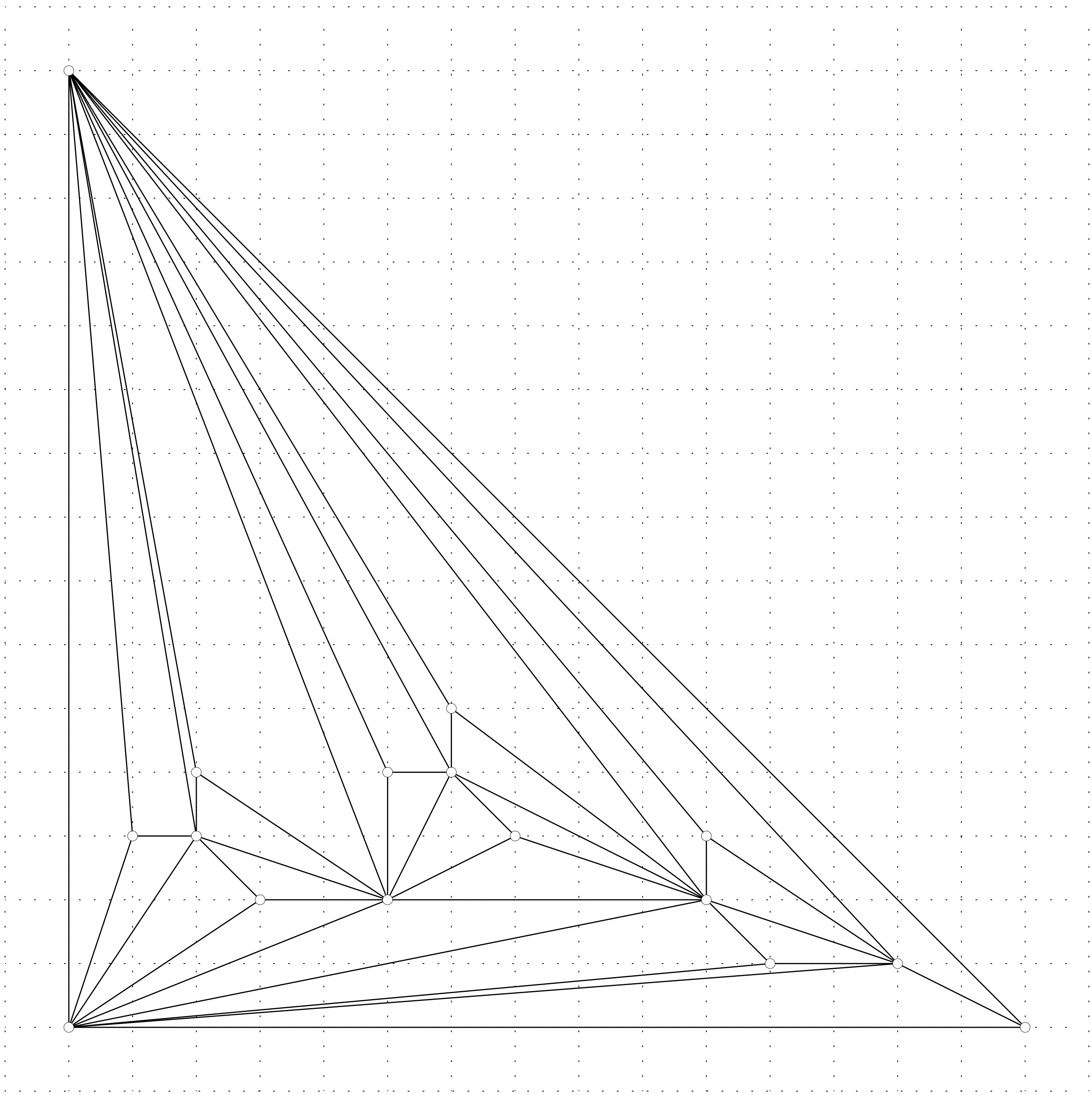

Figure 4: Grid Embedding

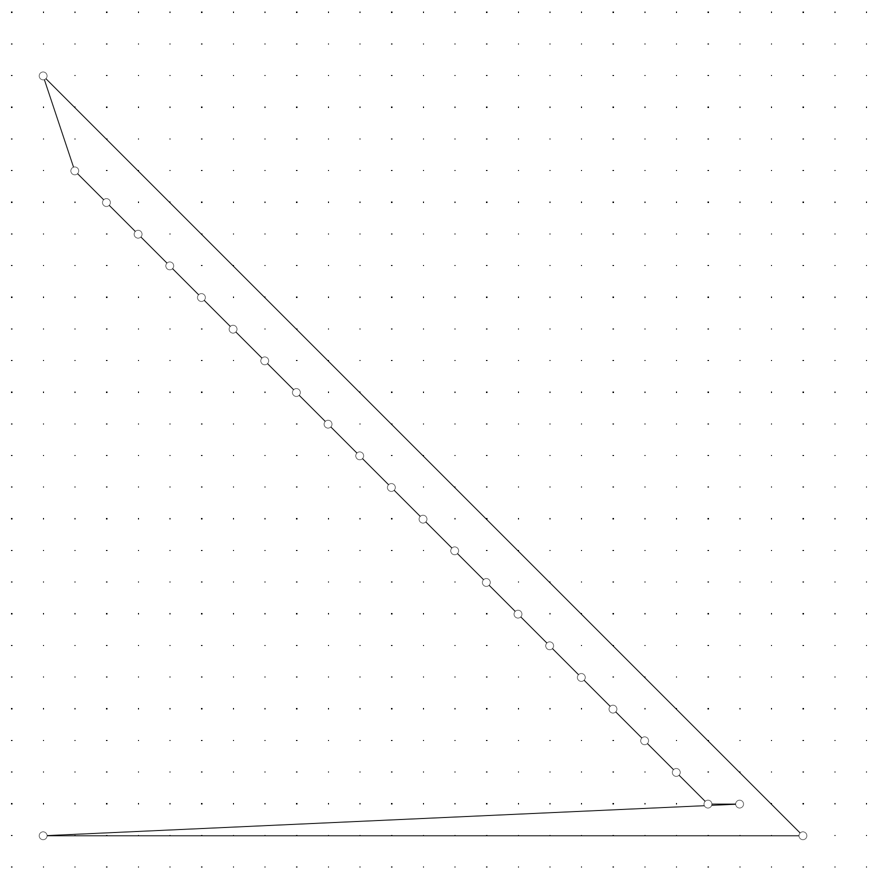

Figure 5: Grid Embedding of a Cycle

The algorithm of Schnyder has other advantages and uses barycentric coordinates to obtain a planar representation of the graph in 3-space.

5 A Statistical Method

Suppose again that you're given a graph (say as an incidence or adjacency matrix) and you're asked to draw it. How should it be drawn? Suppose also that the graph is nonplanar, it is rigid (i.e., has the trivial automorphism group) and crossings are irrelevant. Then the topics discussed so far are of no use. On the other hand we still require that the drawing be representative of the graph in the sense that it effectively conveys the structure of the graph. One possibility is to consider distance in the graph where every edge has unit length. For example, we could insist that the euclidean distance between any two vertices be the same as their distance in the graph. But this is a bit unreasonable since such a layout exists only for paths. Alternatively we suppose that any layout which approximates the distances is acceptable. Specifically, for a graph G with vertices $v_1, v_2, \ldots, v_n$ let $d(v_i, v_j)$ denote the distance in G from v_i to v_j. We desire one point $x_i \in \Re^k$ for each vertex v_i so that the sum $\sum_{i,j}(d(v_i, v_j) - \|x_i - x_j\|)^2$ is minimized. Although the case $k = 2$ may at first seem more natural (a tendency we must unlearn), in general better representations can be obtained by approximating the distances in higher dimensional euclidean spaces. The drawings shown in Figures 6 and 8 were obtained by applying this approach. This approach worked well for many other graphs including paths, cycles, trees, cylinders, grids in both 2 and 3 dimensions, hypercubes, prisms, and the Platonic solids..

This problem is really a special case of a data analysis problem which is well-known to statisticians. (The description given here is somewhat oversimplified.) We are given a dissimilarity matrix $\Delta = (\delta_{ij})_{i,j=1,\ldots,n}$ for n objects $1, \ldots, n$, and it is desired to represent each object i by a point x_i in $\Re^k$ in such a way that the euclidean distances $\|x_i - x_j\|$ approximate the dissimilarities δ_{ij} $(i, j = 1, \ldots, n)$. Since the dissimilarity matrix Δ is derived from actual measurements, it is subject to various types of errors; for example, we could have innaccurate data, missing data, repeated measurements or Δ could be asymmetric. Over the last 20 to 30 years a large literature of mathematical techniques has been developed to handle these complications. Basically, the problem is solved using a technique called multidimensional scaling (MDS). A very nice and very readable introduction to MDS can be found in [19].

6 Implementation and Experimental Results

There are many tools for studying graphs and drawing graphs in a variety of ways, for example see [14]. This section contains an overview of the special software we used for our experiments. All of the programs mentioned here are described in detail in the papers cited below, and so the purpose of this overview is only to identify the tools we used for our investigations. These programs were developed at Bellcore using the X Window System, which ensures portability across a wide variety of workstations and personal computers. We use NETPAD (see [12] and [13]) to construct the graph with an initial embedding in the plane. XGvis [5] is used to construct the MDS embedding in $\Re^d$, and we view the embedding using XGobi (see [3] and [11]).

6.1 NETPAD

NETPAD is a UNIX/X11-based graphics interface experimenting with graphs. It provides the user with a set of elementary functions for manipulating graphs (such as create, delete, select and move) that may be executed simply by pointing and clicking with the mouse. The user may also select items from menus (e.g., change color, set node type, set style of links, scale the graph) and and execute network algorithms. Thease features are remarkably useful even for displaying, printing and redrawing small graphs. See the documentation [12], [13] for more details. Figure 7 contains a sample NETPAD window.

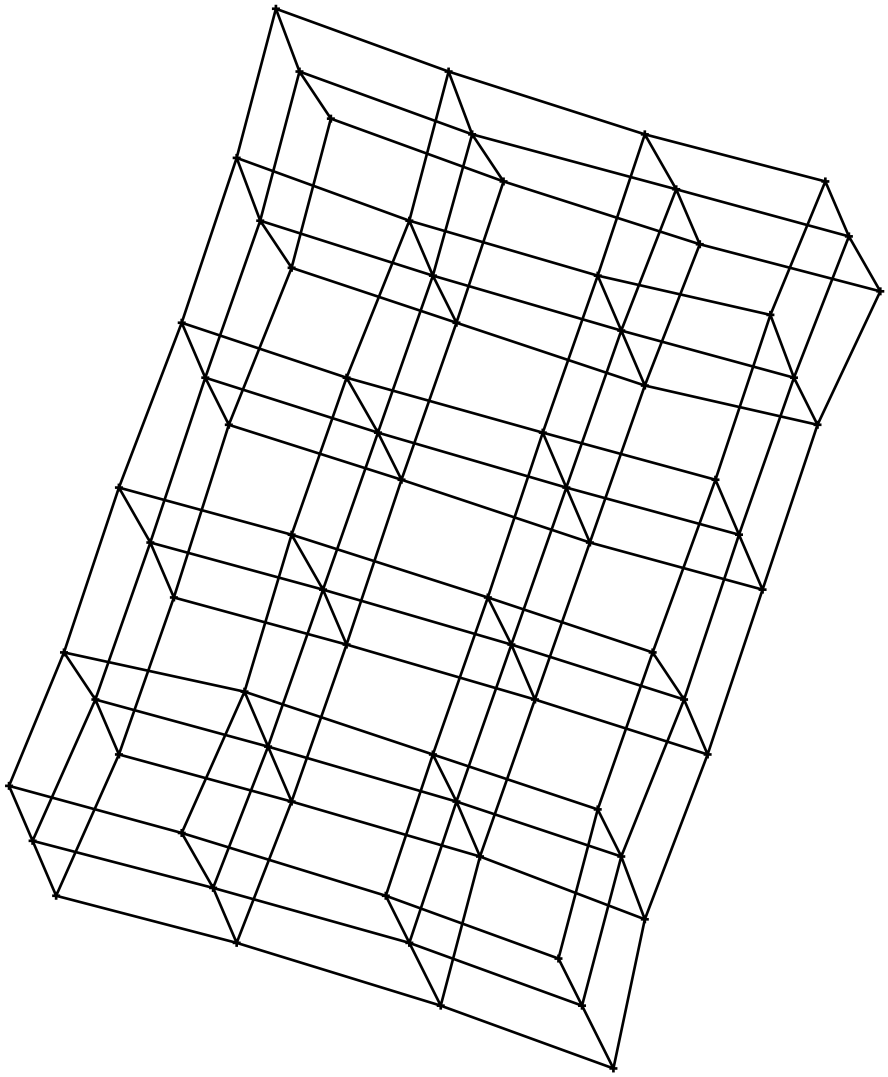

Figure 6: An MDS Embedding

For our investigations with higher dimensional drawing, once the graph is constructed a distance matrix based on shortest path distances is constructed and a function call is made to XGvis. This distance matrix is the dissimilarity matrix used as input to XGvis.

6.2 XGvis

XGvis, a software system which accepts diverse input data, develops a dissimilarity matrix from the data, and then calculates a new representation for the dissimilarity information. XGvis finds this new representation, usually of lower dimension than the original data, by the iterative use of multidimensional scaling.

The applications for such a tool extend beyond graph representations. For example, in survey data on customer satisfaction with telecommunications services, it is often useful to find groups of services that are viewed similarly by customers. To do this, the data are first put into a matrix of pairwise dissimilarities between services, and then a simpler (usually lower-dimensional) representation for the dissimilarity matrix is sought. If a three-dimensional representation is found, then each service is represented by a point in $\Re^3$; a pair of services is found to be similar precisely when the corresponding two points are close together in the embedding. The mathematics and optimization associated with this approach was described briefly in Section 5, and a more detailed explainatin can be found in [5].

6.3 Xgobi

The scope of XGobi also extends beyond this paper. Basically, XGobi is a dynamic graphics program with a link to the S data analysis environment. Xgobi is designed to be used as a complement to the existing analytical tools by making it possible to use each program's unique functions on the same data. For higher dimensional viewing it uses the grand tour to dispay a dynamic sequence (i.e. a animation) of two-dimensional projections of higher-dimensional data. The grand tour is used to quickly uncover structure. The guidance system for the grand tour is based on user-controlled motion and on steerage along the gradients of functions of interest of each two-dimensional projection. One example of such a function would be a function which is sensitive to clumpiness or clustering in the underlying data. Xgobi's guidance system increases the chances that the tour quickly exposes the structure in the data. XGobi's functionality also includes 3-dimensional rotation, the ability to distinguish groups of points with a method called brushing, either in one window at a time or using two or more linked windows, and moving and reshaping of the data in the plot region. It has proved to be a remarkably powerful tool for exploratory data analysis.

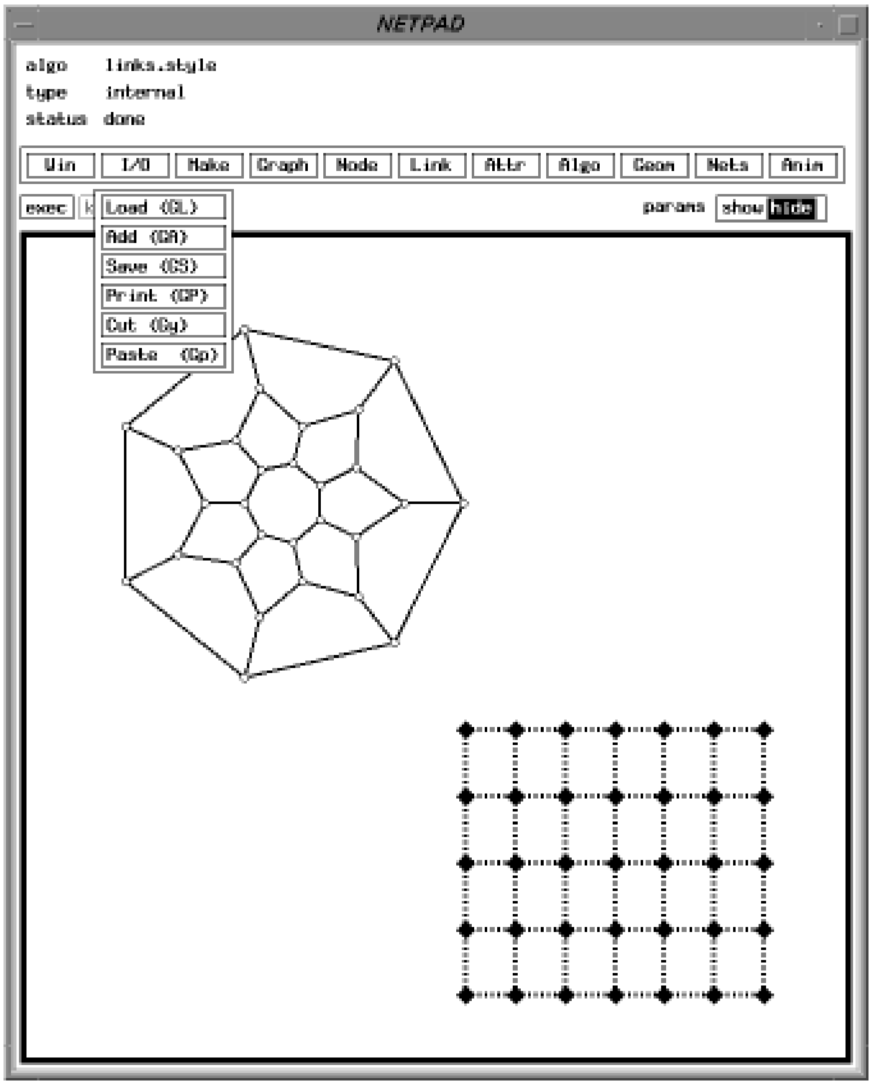

Figure 7: NETPAD Window

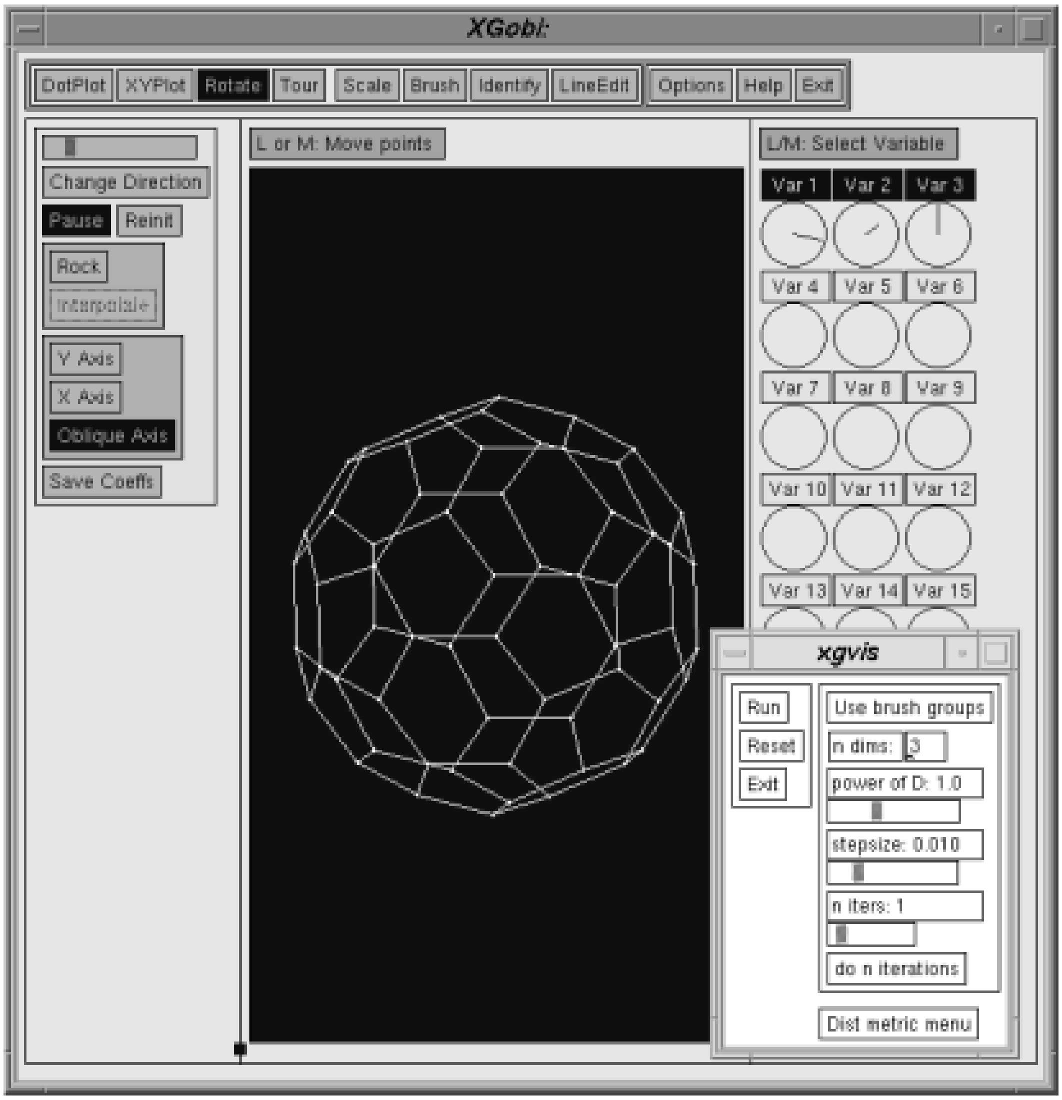

Figure 8: XGobi and XGvis Windows

7 Closing Remarks

We have provided an overview of various methods for drawing graphs in higher dimensions. In particular, we discuss three different types of spring embeddings in Sections 2.2, 2.3 and 5 illustrating connections between graph drawings and notions involving convex polyhedra, connectivity and statistics. It is clear that some areas such as symmetry (Section 3) and grid embeddings (Section 4) deserve a lot more attention. Our discussions and experiments demonstrate that graph theory no longer needs to be limited to a flat world. Theorems and algorithms about graphs in the plane can and should be generalized to higher dimensions.

References

[1] D. Bienstock and N. Dean, New results on rectilinear crossing numbers and plane embeddings, *J. Graph Theory* **16** (1992) 389–398.

[2] J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications.* American Elsevier, New York (1976).

[3] A. Buja, D. Cook and D. F. Swayne, User's Manual for Xgobi, a Dynamic Graphics Program for Data Analysis Implemented in the X Window System (Version 3), *Bellcore Technical Memorandum* (1994).

[4] A. Buja, J. Cabrera and D. Cook, Projection Pursuit Indices based on Expansions with Orthonormal Functions, *J. Computational and Graphical Statistics* **2** (1993) 225–250.

[5] A. Buja, N. Dean, M. L. Littman and D. F. Swayne, Visualizing the Embedding of Objects in Euclidean Space, *Bellcore Technical Memorandum* TM ARH-021761 (1992).

[6] J. Cheriyan and J. H. Reif, Directed s-t numberings, rubber bands, and testing digraph k-vertex-connectivity, *Proc. 3rd ACM-SIAM SODA* (1992) 335–344.

[7] K. B. Chilakamarri, A characterization in Z^n of finite unit-distance graphs in $\Re^n$, to appear.

[8] K. B. Chilakamarri, N. Dean and M. L. Littman, Three-dimensional Tutte embedding, *Congressus Numerantium*, to appear.

[9] M. Chrobak and T. H. Payne, A linear time algorithm for drawing a planar graph on a grid, manuscript (1988).

[10] M. Chrobak, personal communication.

[11] D. Cook and D. F. Swayne, Xgobi: A Dynamic Graphics Program Implemented in X with a Link to S, *Proceedings of the 22nd Symposium on the Interface*, Springer/Verlag (1990).

[12] N. Dean, C. L. Monma and M. Mevenkamp, NETPAD Version 1 - User's Guide, *Bellcore Technical Memorandum* TM ARH-022264 (1992).

[13] N. Dean, C. L. Monma and M. Mevenkamp, NETPAD: An interactive graphics system for network modeling and optimization, *Computer Science and Operations Research*, Pergamon Press (1992) 231–243.

[14] N. Dean and G. E. Shannon (eds.), *Computational Support for Discrete Mathematics*, American Mathematical Society (1994).

[15] P. Eades and R. Tamassia, Algorithms for drawing graphs: An annotated bibliography, *Technical Report 82*, Department of Computer Science, Univ. of Queensland, Australia (1987).

[16] P. Erdős, F. Harary and W. T. Tutte, On the dimension of a graph, *Mathematika* (1961) **12** 118–122.

[17] H. de Fraysseix, J. Pach and R. Pollack, Small sets supporting Fáry embeddings of planar graphs. *Proc. 20th Ann. ACM Symp. on Theory of Computing* (1988) 426-433.

[18] A. Itai and M. Rodeh, The multi-tree approach to reliability in distributed networks, *Information and Computation* **79** (1988) 43–59.

[19] J. B. Kruskal and M. Wish, *Multidimensional Scaling*, Sage Publications, Beverly Hills, 1978.

[20] F. T. Leighton, *Complexity Issues in VLSI*, The MIT Press.

[21] A. Lempel, S. Even and I. Cederbaum, An algorithm for planarity testing of graphs. In *Theory of Graphs: Internat. Symp.: Rome*, P. Rosensteil, Ed., Gordon and Breach, New York (1966) 215–232.

[22] N. Linial, L. Lovasz and A. Wigderson, Rubber bands, convex embeddings and graph connectivity, *Combinatorica* **8** (1988) 91–102.

[23] R. J. Lipton, S. C. North and J. S. Sandberg, A method for drawing graphs, *STOC* (1985) 153–160.

[24] L. Moser and W. Moser, Problem and solution P10, *Canadian Math. Bull.* **4** 187–189.

[25] W. Schnyder, Embedding planar graphs on the grid, *Proc. of the First Annual ACM-SIAM Symp. on Discrete Algorithms* (1989) 138-148.

[26] E. Steinitz, Polyheder und Raumeinteilungen. *Enzykl. math. Wiss.* **3** (Geometrie), Part 3AB **12** (1922) 1-139.

[27] W. T. Tutte, How to draw a graph, *Proc. London Math. Soc.* **13** (1963) 743-767.

[28] K. Wagner, Bermerkungen zum Vierfarbenproblem, *Jber. Deutsch. Math. Verein* **46** (1936) 26-32.