
Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 9, no. 3, pp. 301–304 (2005)

Special Issue on Selected Papers from the Twelfth International Symposium on Graph Drawing, GD 2004

Guest Editors' Foreword

Emden R. Gansner
AT&T Labs - Research
http://www.research.att.com/info/erg
erg@research.att.com

János Pach
City College of New York and The Courant Institute
http://www.math.nyu.edu/ pach/
pach@CIMS.nyu.edu

Continuing a tradition, this special issue presents journal versions of some of the best papers given at the *Twelfth International Symposium on Graph Drawing*, which was held from September 29 to October 2, 2004 at City College, CUNY, in New York. As is apparent from the ongoing symposia programs, graph drawing remains an active and exciting research area, with intriguing problems in pure mathematics as well as real-world applications. This diversity is reflected in the papers appearing here. It should be noted, though, that with the growing importance of information visualization, of which graph drawing can be considered a part, one year's pure mathematics is next year's application.

Sugiyama's algorithm is the standard technique for drawing directed, especially hierarchical, graphs and networks. The algorithm consists of a sequence of phases, each adding additional information to the output of the previous phase. Since the solution of each of these is largely independent of the other phases, one can pick a solution for a particular phase that best fits certain constraints on the layout. Or, with increasingly larger graphs being drawn, one can consider how a particular phase or phases can be made more efficient, without paying too much in reduced quality of the layout. The paper "An Efficient Implementation of Sugiyama's Algorithm for Layered Graph Drawing," by M. Eiglsperger, M. Siebenhaller, and M. Kaufmann, falls into the second category. Traditionally, after nodes have been placed in ranks, edges crossing multiple ranks are divided into chains of edges, with each edge connecting two adjacent ranks. The algorithm uses this significantly larger graph for the remaining phases, including the arrangement of nodes within ranks to reduce edge crossings. For large graphs, this division leads to unsatisfactory performance. The paper considers what would happen if, instead of dividing long edges into arbitrarily many pieces, one splits the edge into three pieces, a small head and tail piece and one long, vertical piece. The paper shows that this change improves the performance by an order of magnitude. More surprisingly, it shows that this is achieved without incurring a single new edge crossing.

One application of graph drawing concerns communication within sensor networks, as discussed in the paper "Distributed Graph Layout for Sensor Networks," by C. Gotsman and Y. Koren. These devices typically have limited physical resources, which constrains their communication and computational capabilities. One effect of this is that a sensor may only know how far it is from a set of neighboring sensors. To further complicate matters, the sensor network may be dynamic, with sensors being moved, added, or removed, and noise is unavoidable. For various reasons, it is important that each sensor knows its position relative to all of the other sensors, under the conditions described above. This is basically a graph drawing problem. If all pairs of distances are known, it can be handled by stress minimization using classical multidimensional scaling (MDS). Here, with incomplete distance information, it is possible for a solution to have part of the graph folded over on itself. Gotsman and Koren employ spectral techniques to get an initial solution without folds. To get the final layout, they apply stress minimization to the initial layout, but employ the more effective majorization technique rather than the more standard use of gradient descent.

Sensor networks provide one class of dynamic graphs. In general, the various abstract graphs associated with modern telecommunications tend to be very dynamic. Social networks are also dynamic, though over a larger time scale. The importance of analyzing how graphs in these and other areas evolve with time has led to much recent work into dynamic graph drawing. The tension here is between providing layouts in which a small change in the graph produces a small change in the layout, thereby helping to maintain the user's context, versus having each layout optimized to best display the graph's structure. This trade-off leads naturally to the question of how to simultaneously draw two graphs on the same vertex set. It is this problem that C. Erten and S. Kobourov consider in their paper "Simultaneous Embeddings of Planar Graphs with Few Bends." They show that it is too much to ask for planar drawings using line segments. If one allows at most three bends per edge, however, they are able to provide a linear-time algorithm which constructs a simultaneous embedding with the vertices placed on the $O(n^2) \times O(n^2)$ grid, where n is the number of vertices. When both graphs are trees, a case arising in evolutionary biology, they reduce the number of bends needed to one.

Radial drawings, in which vertices are positioned on concentric circles, form another style of graph drawing that has found wide application. These have been used effectively in viewing policy networks, file systems, protein-protein interaction diagrams, and telecommunication networks. They are particularly relevant when part of the graph plays a more central role, which is reflected in a more central position in the layout. Such layouts can be constructed quickly, though edge crossings can be a problem. As these are the *bêtes noires* of graph drawing, one would like radial layouts to reduce them. In particular, a planar graph should be drawn with no crossings. It is simple to construct such radial drawings, as long as one does not care how many circles are used, but this weakens the impact of the layout. One would like the planar layout to use only a small number of circles. This is where E. Di Giacomo, W. Didimo, G. Liotta, and H. Meijer step in with their paper "Computing Radial Drawings on the Minimum Number of Circles." They characterize those planar graphs which have a radial drawing using at most k circles, and from this derive a polynomial-time algorithm to construct a crossing-free radial layout of a planar graph with the minimum number of circles.

In many domains, the associated graphs have additional structure, in which the nodes are partitioned into disjoint subsets, each of which may be further partitioned. Consider, for example, the domains induced by IP addresses, or any containment hierarchy. Such graphs are called clustered graphs, with the subsets forming the clusters. By convention, drawings of clustered graphs should position the nodes within clusters nearby each other. Often the clusters have a boundary drawn about them to further emphasize this extra structure, placing each cluster in its own region of the plane. With graphs, the reason to minimize edge crossings is to avoid artifacts in the drawing which confuse the eye. For the same reason, in clustered graphs, one can ask in addition that edges cross into regions only as required by the graph structure. Clustered graphs having such drawings are known as c-planar. Connectivity also has a cluster analogue, in

which a graph is connected if the subgraph induced by each cluster is. Testing for and drawing c-planar graphs has been solved for connected graphs, but is still open for unconnected graphs. The paper "Clustering Cycles into Cycles of Clusters," by P. Cortese, G. Di Battista, M. Patrignani, and M. Pizzonia, considers this problem for highly unconnected graphs. They consider the special case of k-cluster cycles, in which the underlying graph is a cycle and the graph has a coloring using k colors. The authors show that c-planarity testing and drawing can be done efficiently for these graphs, and extend these results to the recursive structures in which cycles of clusters are clustered into another cycle of clusters, to arbitrary depth.

The papers presented in this special issue give a good idea of the variety and vitality found in graph drawing. The solutions draw on graph theory, computational geometry, algorithms, and statistics, among other disciplines, and the results have both an inherent beauty and significant applications.