Preface

Drawings are an attractive and effective way of conveying information. *Graph Drawing* includes all aspects of visualizing structural relations between objects. The range of topics extends from graph algorithms, graph theory, geometry and topology, to visual languages, visual perception, and information visualization, and to computer-human interaction and graphics design. Clearly, the design of appropriate drawings is a complex and costly task where automation is urgently required.

The automated generation of graph drawings has important applications in many areas of computer science, such as compilers, data bases, software engineering, VLSI and network design, and graphical interfaces. Applications in other areas include graphical data analysis (e.g. in all fields of engineering, biology, or social sciences) and the visualization of information in general (e.g. by flow charts, schematic maps, or all kinds of diagrams).

The purpose of this book is to give an overview of the state of the art in graph drawing. It concentrates on algorithmic aspects, with an emphasis on interesting visualization problems with elegant solution methods. Each chapter provides a survey of some part of the field; in addition some selected results are described in more detail. This approach should make the book suitable for a first introduction as well as a good basis for an advanced course, where it may be supplemented by other sources. There is no claim of completeness – graph drawing is a very dynamic area – so the reader should be aware of the possibility that further progress might have been made since the publication of this book. There is also a chance that we may have failed to notice some subjects, since the necessity of drawing graphs arises in so many different areas.

The rapid growth of graph drawing as a field has caused some inconsistencies in terminology: terms like "drawing", "layout", "representation", or "model" are often used with different meanings. The authors have tried to achieve consistent notation; this has not always been possible without breaking with existing conventions, and we apologize for all remaining inconsistencies.

The book arose from a seminar for young computer scientists. The idea of the "GI Research Seminars" is to provide young researchers with the opportunity to gain insight into a new, relevant, and interesting area of computer

science. The topics were chosen and prepared by the organizers. Lectures on the topics were then presented by the young researchers at the research seminar, which took place in April 1999 at Schloß Dagstuhl. Based on the presentations and discussions during the seminar, each chapter was elaborated by a team of authors, and carefully reviewed by other participants of the seminar. In addition, we obtained expert advice from Therese Biedl and David Wood. They provided detailed comments and improvements on several chapters in a preliminary version of the book.

The material covered in this book can be organized in various ways. However, a strict separation of topics according to, e.g., models, representations, or methods, would inevitably lead to an artificial distinction between topics that are in fact closely related with respect to some other aspect. Therefore, we decided to do without a strict partition into chapters under a single category of distinction, but rather provide cross-pointers where appropriate.

In the first chapter, Rudolf Fleischer and Colin Hirsch review some *application areas*. They discuss the traditional applications of graph drawing like ER-diagrams or software engineering, but also some of the areas that are less related to computer science like social networks or workflow. From those applications, different tasks and basic techniques are derived such that the reader can get some motivation to learn of the methods to solve the tasks.

We then start from a more graph-theoretic point of view. *Planarity* is a classical topic in graph theory and algorithms, and an important and central aspect in graph drawing as well. René Weiskircher reviews basic algorithms related to planar graphs (planarity tests, *st*-orderings, etc.). He then turns to simple and advanced drawing algorithms for straight-line and related representations of planar graphs.

In the following chapter, some further *special classes of graphs* are considered by Matthias Müller-Hannemann. Drawing algorithms for trees and series-parallel graphs typically use their recursive structure. A closely related issue is the drawing of order diagrams and lattices, which is summarized as well. Although at first sight, this topic seems to be only of special interest, the algorithms presented have direct applications since many structures to be visualized are just trees or series-parallel.

In the chapter on methods based on *physical analogies*, Ulrik Brandes gives an overview of the ideas behind the so-called spring embedder and its variants. These methods work by iterative improvement. Related methods to compute a minimum of the objective function directly are included as well. In the last subsection, the reader gets an idea of the wide field of possible applications of the underlying principle which ranges from clustering and 3D to dynamics and constraints.

A classical topic in graph drawing which cannot be missed in a review like this is an approach supporting *layered drawings*. This method is mainly used to display temporal structures like workflow or other unidirectional dependencies. Several interesting methodological questions like the maximal

acyclic subgraph problem and the level-wise crossing minimization problem are discussed in this chapter by Oliver Bastert and Christian Matuszewski.

A long chapter is devoted to *orthogonal layouts*, written by Markus Eiglsperger, Sándor P. Fekete, and Gunnar W. Klau. Historically, the first methods for orthogonal layouts were developed in the context of VLSI routing and placement. Changing some of the criteria gave room for improvements and variations like visibility representations. After the review of some heuristics for planar and nonplanar graphs, a very elegant flow-based approach to solve the bend-minimization problem for planar graphs is considered in the second part of the chapter. The final part is devoted to compaction, i.e., the post-processing phase where the components of the layout are squeezed together to save area, edge length, or bends. Here again, VLSI methods are reviewed and new LP-based methods are presented that solve the compaction problem regarding some specific criteria to optimality.

The methods described in chapters 1–5 are fundamental. The following are more advanced and mostly arise from application demands such as labeling, clustering and hierarchies, dynamics and interactiveness, or 3D.

The generalization from drawing in the plane to *3D graph drawing* is discussed by Britta Landgraf. Various representations are covered. Starting from force-directed methods and layered approaches such as cone trees, she reviews the most important techniques of orthogonal routing in 3D in more detail and ends with the discussion on finding the best point from which to view a three-dimensional structure.

Handling large graphs is an important problem where some new methods are needed. Sabine Cornelsen and Ralf Brockenauer present some of the common *clustering techniques* like partitions and structural clusterings. Furthermore, hierarchical graphs are discussed with the focus on planar drawings and on the concept of compound graphs as well. Moreover, it is shown how force-directed methods can be applied to visualize clusters.

Dynamic graph drawing is a very recent and relevant topic. Jürgen Branke highlights many concepts in this area, like maintaining the mental map and support of dynamics in orthogonal structures and force-directed approaches.

At first glance, *labeling* is a side topic in graph drawing. However, labels arise in nearly all practical applications, and the labeling problem is highly nontrivial. Gabriele Neyer reviews different labeling models and most important methods. Most of them come from the fields of computational geometry, cartography, and optimization. The various aspects considered include the relation to satisfyability problems, sliding labels, and the combination with compaction.

In various aspects graph drawing is motivated by the relevance of visualizing relational data in many field of applications. Therefore, *software tools* for drawing graphs and *algorithms libraries* are an important issue in graph drawing. However, some difficulties about this topic prevented us from making it a regular chapter. Many graph drawing tools and libraries have been

developed over the last few years, but easy access to the tools, stability, and support are often not guaranteed. Typically, software developed in academia cannot have the pretension of being "made for eternity". Moreover, it is hard to get an overview of the graph drawing tools available, or even reliable information about their actual abilities. On the other hand, we thought it would be useful to share what we knew of some existing systems. In the appendix, Thomas Willhalm provides an overview of some of the most common systems for graph drawing.

Finally, it is our pleasure to thank all the people whose contribution has made this book possible. First and foremost, these are the authors of the chapters, who not only put a huge amount of work into their own chapters, but also carefully reviewed other chapters. Special thanks go to Ulrik Brandes, who supported this project from the very beginning in many ways. Not only did he do a lot of administrative work in preparation of the research seminar, but he also inspired the choice of topics and supported the collection of relevant material. Last but not least, he handled most of the technical parts of the editing process. Finally, we would like to express our gratitude to the external experts Therese Biedl and David Wood, who did us the favor of reviewing several chapters in a preliminary version of the book.

January 2001

Michael Kaufmann
Dorothea Wagner