Layout of Graph Visualizations

Ulrik Brandes

Dissertation der Universität Konstanz
Fakultät für Mathematik und Informatik
1999

Teile dieser Arbeit wurden bereits in Brandes und Wagner (1997, 1998a,b, 1999), Brandes, Kenis, Raab, Schneider und Wagner (1999) bzw. in den *Konstanzer Schriften in Mathematik und Informatik* Nr. 33, 40, 60, 62, 69 und 85 veröffentlicht.

1. Referent: Prof. Dr. Dorothea Wagner, Universität Konstanz
2. Referent: Prof. Dr. Michael Kaufmann, Universität Tübingen

Tag der mündlichen Prüfung: 10. Juni 1999

Preface

My first memory of the subject of this thesis is my own ignorance. In September 1995, Dorothea Wagner had arranged for the newer members of her working group to attend a symposium held in Passau. The theme of the event was manifested in its title, *Graph Drawing*, and I wasn't quite expecting to plunge into my new area of interest. After all, mathematicians and computer scientists were going to speak about pictures of binary relations.

Things turned out different from what I expected, obviously. And it is my sincere hope that some of the excitement I find in graph visualization has found its way into this thesis.

Taking us to this particular conference is only one good Dorothea Wagner did us. It is with great pleasure and deep gratitude that I acknowledge her supervision and support. She would teach me many different things (sometimes without even knowing), give me directions, listen to my outlandish ideas, let me go to conferences, introduce me to amazing people, and do all the other things a student can hope for, but never expect from his supervisor. Hopefully I can return at least some of this someday.

Many people have contributed directly or indirectly to this thesis and I apologize for mentioning only a few. Thanks to Michael Kaufmann, who readily agreed to be a thesis reviewer, and Marc H. Scholl and Volker Schneider for joining my examination committee.

From *Graph Drawing '95* I returned to Konstanz with the urge to find graphs that would be interesting to visualize, and I vaguely recalled a friend mentioning "political networks" or such (credit goes to Natascha Füchtner). To make a long story short, by June 1996 an interdisciplinary project group was set up, starting to explore the visualization of social networks.[1] Working together with Patrick Kenis, Jörg Raab, Volker Schneider, and Dorothea Wagner was an invaluable experience from which I profited both professionally and personally.

In similar, and sometimes also completely different, ways I benefited from

[1]Financially supported by the *Ausschuß für Forschungsfragen der Universität Konstanz*.

sharing an office with Annegret Liebers. Thanks for sustaining, proofreading part of this thesis, and never-ending food supply.

The embarrassment of writing about visualization without visualizations was avoided only with the much appreciated help of Marco Gaertler, Michael Güdemann, Vanessa Kääb, Andres Löh, Frank Müller, and Thomas Willhalm, who combined stand for most of the implementation work.

Joining the newly created Computer Science group at the University of Konstanz in October 1994, I was fortunate to become part of a stimulating and supportive environment. While I thank all those contributing to it, I blame my colleagues and friends Dieter Gluche and Torsten Grust for mornings lost in hangover. Our weekly "E-Seminar" was an important institution in which it was *expressis verbis* allowed to talk about thesis related topics – soccer, for instance.

Standing in for all those people that have been important during this phase of my life, special thanks go to my family, Angelika, Klaus, and Timo Brandes, and to Biene for being *mein Mensch*.

Deutsche Zusammenfassung

Visualisierung ist ein attraktives und effektives Mittel sowohl der Präsentation als auch der Exploration von Daten. In beiden Fällen ist das oberste Ziel die getreue Darstellung der durch die Daten repräsentierten Information in leicht verständlicher Form.

Handelt es sich um relationale Daten, besteht der wesentliche Informationsgehalt in der Struktur der Beziehungen. Die entscheidende Aufgabe bei der Visualisierung von Graphen, also der graphischen Darstellung binärer Relationen, ist daher der Abgleich der räumlichen Anordnung innerhalb des Diagramms mit den strukturellen Eigenschaften des Graphen. Dieser Schritt und sein Ergebnis werden auch das Layout der Visualisierung genannt. Dabei sind so komplexe Abhängigkeiten zu berücksichtigen, daß aussagekräftige, auf objektivierten Kriterien beruhende Visualisierungen durch eine algorithmische Behandlung des Layoutproblems überhaupt erst möglich werden.

Layout von Graphenvisualisierungen ist damit insbesondere ein Problem der angewandten Informatik und diskreten Mathematik, dessen Bedeutung zusammen mit dem Bedarf an Mitteln zur Analyse und Vermittlung komplexer Informationen wächst, bei dem jedoch sowohl die Modellbildung (Beschreibung geeigneter Darstellungen) als auch die Problemlösung (automatische Erzeugung geeigneter Darstellungen) zahlreiche ungeklärte Fragen aufwerfen. Wichtigstes Ergebnis dieser Arbeit sind ein vereinheitlichender Formalismus für die Modellbildung, sowie mehrere darauf beruhende Modelle und Algorithmen. Sie gliedert sich in drei wesentliche Abschnitte.

Zunächst wird Layout in Kapitel 2 als Optimierungsproblem hergeleitet. Gute räumliche Anordnungen in Diagrammen entsprechen dabei den Minima von Bewertungsfunktionen, die aus gewichteten Einzelkriterien zusammengesetzt sind. Dieser Ansatz ist sehr allgemein und flexibel, da beliebige Einzelkriterien wie hinreichender paarweiser Abstand bestimmter graphischer Elemente, bevorzugte Positionen, usw. auf unterschiedliche Weise kombiniert werden können. Darüber hinaus kann gezeigt werden, daß sich das Layout dynamischer, d.h. mit der Zeit verändlicher, Graphen durch Hinzunahme von Stabilitätskriterien uniform auf das statische Problem zurückführen läßt.

In Kapitel 3 wird anhand dreier Fallstudien demonstriert, daß sich die Formulierung über Einzelkriterien vor allem zur experimentellen Bestimmung von Bewertungsfunktionen für Graphen aus Anwendungsgebieten eignet, für die noch keine befriedigenden Layoutvorschriften vorliegen. Die untersuchten Graphen repräsentieren Soziale Netzwerke, dynamische WWW-Links und Fahrplangraphen von Zug- und Fährverbindungen. In allen drei Fällen werden neue, auf den spezifischen Informationsgehalt der Graphen zugeschnittene Modelle entwickelt.

Das wesentliche Ergebnis in Kapitel 5 ist schließlich ein Verfahren zum dynamischen Layout von Diagrammen, in denen alle Verbindungslininen achsenparallel verlaufen sollen (wie zum Beispiel in Schaltplänen häufig der Fall). Dazu wird ein wichtiges Verfahren aus der Literatur um die in Kapitel 2 hergeleiteten Stabilitätskriterien erweitert. Da das statische Verfahren auf Netzwerkflüssen beruht, können diese Kriterien als Zielwerten für den Fluß gedeutet werden. Durch Modifikation des Netzwerkes können optimale dynamische Layouts wie im statischen Fall effizient über Flüsse minimaler Kosten bestimmt werden.

Wo dies geboten erschien, sind aus der Literatur bekannte Verfahren und Ergebnisse, zum Teil in neuer Darstellung, eingefügt. Alle vorgestellten Ansätze wurden implementiert, und zahlreiche Abbildungen zeigen die erzielten Resultate.

Contents

Chapter 1

Introduction

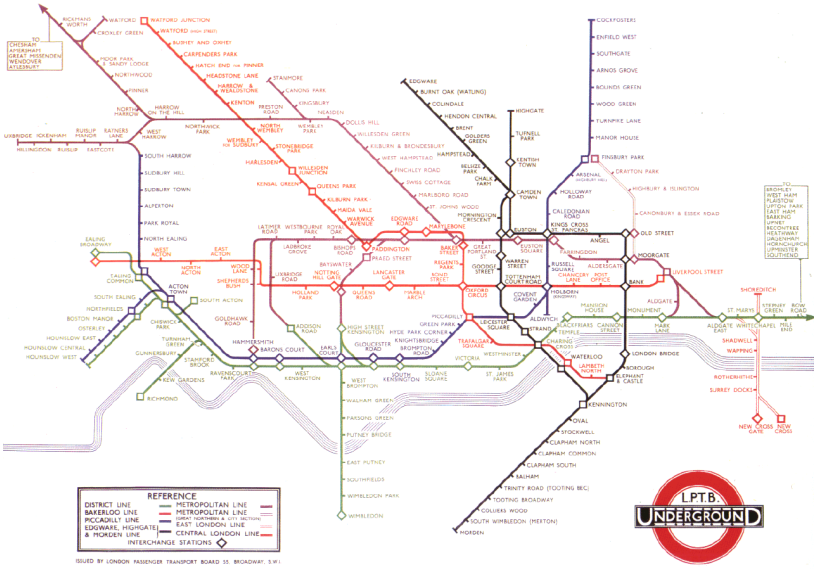
When lost in an unknown metropolitan area, many simply enter the next metro station and return to a known place. The ease of applying this strategy is partly due to the schematic maps featured by metros all over the world. Diagrams like in Figure 1.1(a) make it easy to find a suitable route, even without speaking the local language.

The effectiveness of their simplistic, yet information dense design becomes evident when compared to geographically more accurate maps as in Figure 1.1(b). Note that in both maps train routes are coded using color (matching lines even with the same color) and connecting stations are coded using a special shape.

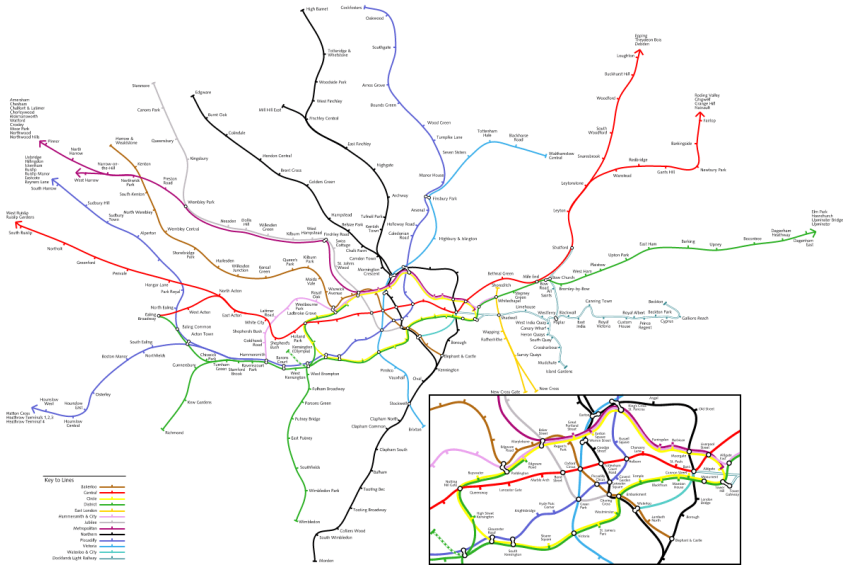
The decisive difference of these two maps is their layout. Even though only horizontal, vertical, and diagonal lines are used in order to reduce visual complexity, the geographically inaccurate schematic map resembles the true courses fairly well. Equidistant stops along straight segments reflect the way passengers measure distances, namely by counting stops. Instead of using an additional window, the dense and highly frequented center is enlarged with respect to the surroundings.

The purpose of schematic railway maps, and hence the objective of their layout, is clearly not to display the exact geographical location of stations, but to ease the inference of travel information. The essence of this information is given by the graph of existing connections, *i.e.* the way in which stations are linked by railway lines. The maps in Figure 1.1 are therefore examples of graph visualizations, and it is their layout that dominates their utility.

Visualization of graphs is conventionally called *graph drawing* (Di Battista *et al.*, 1994, 1999; Brandenburg *et al.*, 1997). While this sounds like an artistic endeavor, aesthetically pleasing pictures are not our primary goal. From an information visualization perspective, graph drawing aims at effective



(a) First schematic map of underground railway lines (designed by Harry Beck, first published in 1933 by London Transport). Garland (1994) reviews the history of the design seminal for the now familiar look of public transportation maps



(b) Geographically accurate map of London's underground railway lines as of 1998 (from the Jubilee Line Extension web site `http://www.jle.lul.co.uk/misc/tubeinfo/alternativemap.htm`)

Figure 1.1: Different layout of graph visualizations

communication of the information represented by the graph. This thesis is about automatic layout of graph visualizations. Its scope is delineated in Chapter 2, where graph drawing is embedded in the context of graphical presentation, and a uniform formalism for layout is developed.

Graphs to be visualized arise in different contexts and thus represent different kinds of information. Three case studies, in which layout methods for graphs representing structures as distinct as social networks, links among web pages, and train connections are derived, form the centerpiece of Chapter 3. These methods are based on an analogy to physical systems of objects that are subject to various forces, and they determine positions for graphical primitives directly. Some properties of the classic barycentric approach sharing this physical analogy are outlined in Chapter 4. One of its main drawbacks then motivates the consideration of layout approaches that employ an intermediate step in which angles formed by graphical primitives are determined prior to positioning. Several aspects of such approaches are discussed in Chapter 5, and an extension to graphs that change over time is introduced for a popular schematic layout method that uses only horizontal and vertical lines. Content and contributions of this thesis are reviewed in more detail in Chapter 6.

Chapter 2

A Formal Framework

Aware that the intuitive use of many terms in graph drawing can be rather confusing, we here attempt to provide a framework for layout of graph visualizations that is thoroughly formal. Naturally, the object of interest is defined first.

A *graph* $G = (V, E)$ is a pair of *vertices* $V = \{v_1, \ldots, v_n\}$ and *edges* $E = \{e_1, \ldots, e_m\}$, where each edge $e \in E$ is an ordered or unordered pair of vertices. An ordered pair $e = (u, v) \in V \times V$ is called a *directed edge*, while an unordered pair $e = \{u, v\} \subseteq V$ is called an *undirected edge*. In case $u = v$, e is called a (directed or undirected) *loop*. If E is a multiset, edges with multiplicity greater than one are called *multiple edges*. If we explicitly allow loops or multiple edges, the graph is also called a *multigraph*, otherwise it is a *simple* graph. Speaking of graphs, we usually refer to simple, undirected graphs.

Two vertices of a graph are *adjacent*, if they are connected by an edge. A vertex and an edge are *incident*, if the vertex is part of the edge. Two edges are incident, if they share a vertex. The number of edges incident to a vertex v is called the *degree*, $d_G(v)$, of that vertex. In graphs that contain directed edges, it makes sense to define the *indegree* (*outdegree*) of a vertex to be the number of incident edges that are either not directed or do not have the vertex as their first (second) component. Further graph terminology is defined only when needed.

We are going to derive graph visualization as a form of presenting relational data. Three crucial aspects of graphical presentation of graphs, and information visualization in general, are identified. Since layout is the main difficulty in the design of a graphical presentation of relational data, a uniform formalism for layout specification is provided. General methods for layout computation and some remarks about their implementation conclude the chapter.

2.1 Graph Visualization

The purpose of data presentation is to reveal information buried in a set of data either to the analyst (*exploration*), or to those interested in it (*communication*). A presentation is said to be *effective*, if it conveys the true information in an easily comprehensible, yet precise, way. Since the groundbreaking work of Lambert, Playfair, and others in the 18th century (Tilling, 1975), much effort has been devoted to the development of guidelines for effective presentation of categorical data (Tukey, 1977; Tufte, 1983; Cleveland, 1985, 1993), whereas formal methods for visualizing relational data are still not far from their infants. In this section, some of the more general insights into the technicalities of presenting graphs are reviewed. See also Mackinlay (1986) or Eick (1996).

2.1.1 Forms of Presentation

When relational data is to be explored or communicated, there are three basic forms in which it can be presented to the recipient: *textual*, *tabular*, and *graphical*. Each of these has its own advantages and disadvantages, and ideally they are used in a symbiotic combination (Mahon, 1977).

Textual Presentation. Probably the most precise and flexible, but also a very inconvenient way of presenting a graph. It usually consists of lists of vertices and edges as in Figure 2.1(a).[1] However, further explanations and calculations may accompany the raw data. Since textual presentations are essentially sequential, even core statements are difficult to memorize and to put into context. Best suited for elaborate or tricky explanations.

Tabular Presentation. Typically an adjacency or incidence matrix. *E.g.*, the *adjacency matrix* $A(G) = (a_{u,v})_{u,v \in V}$ of an undirected, simple graph $G = (V, E)$ is defined by

$$a_{uv} = \begin{cases} 1 & \text{if } \{u, v\} \in E \\ 0 & \text{otherwise} \end{cases}$$

whereas its *incidence matrix* $I(G) = (i_{ve})_{v \in V, e \in E}$ has entries

$$i_{ve} = \begin{cases} 1 & \text{if } v \in e \\ 0 & \text{otherwise} \end{cases}$$

[1] Observe that a Wordgraph™ (Garvey, 1998) is not a pure textual presentation, but a two-dimensional spatial alignment of text labels.

As is illustrated in Figure 2.1(b), tabular presentations are more compact than textual ones. The main degrees of freedom are coding of the entries and ordering of rows/columns, to a limited extent allowing to represent underlying structural patterns. It is difficult to represent additional information like, for instance, vertex attributes and more complex structural properties. Best suited for data transmission.

Graphical Presentation. More flexible than tabular presentation and still reasonably compact. Figure 2.1(c) illustrates why graphical presentation is more convenient and appealing. Visual clues are also much easier memorized and contextualized, but offer less flexibility, especially for abstraction. A unique advantage is ability to switch between many different levels of detail without loosing context.

2.1.2 Aspects of Visualization

If relevant information is expressed by honestly generated visual clues, the transformation of data into graphical presentations is called *information visualization*. An ideal visualization would, in the shortest amount of time, reveal to its reader the information, the whole information, and nothing but the information represented by the data. A visualization method should therefore clearly identify the relevant kind of information, define an appropriate mapping of this information to the elements of a graphical presentation, and generate images accordingly. We refer to these three aspects as substance, design, and algorithm, respectively.

Substance. Graphs are used to model relational information in an abstract, to the application of general methods and derivation of general statements amenable way. However, the underlying domain-specific meaning of the graph is the essence of what is to be recognized in a graphical presentation. We call the information that is to be explored or communicated when presenting a particular graph its *substance*. Consequently, a graphical presentation should be prepared in close accordance with the substance to be conveyed. Any open, *i.e.* unspecific, data presentation is either confusing – a "crypto-graphical mystery" (Tufte, 1983, p. 153) – or ambiguous. In both cases, it might even suggest false information.

Substance can be divided into *syntactic* information comprised solely of the graph's structure, no matter what is actually represented, and *semantic* information not captured by the binary relation alone. The case study on social networks in Section 3.2 contains concrete examples for the syntactic and

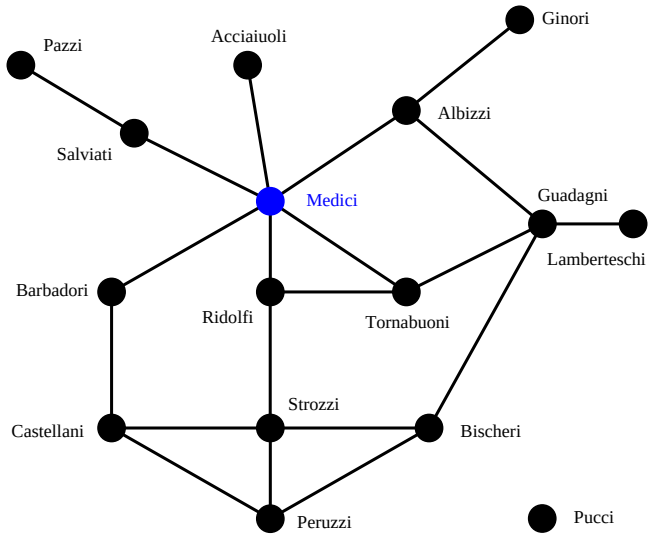
V = { Acciaiuoli, Albizzi, Barbadori, Bischeri, Castellani, Ginori, Guadagni, Lamberteschi, Medici, Pazzi, Peruzzi, Pucci, Ridolfi, Salviati, Strozzi, Tornabuoni }

E = { { Acciaiuoli, Medici }, { Albizzi, Ginori }, { Albizzi, Guadagni }, { Albizzi, Medici }, { Barbadori, Castellani }, { Barbadori, Medici }, { Bischeri, Lamberteschi }, { Bischeri, Peruzzi }, { Bischeri, Strozzi }, { Castellani, Peruzzi }, { Castellani, Strozzi }, { Guadagni, Lamberteschi }, { Guadagni, Tornabuoni }, { Medici, Ridolfi }, { Medici, Salviati }, { Medici, Tornabuoni }, { Pazzi, Salviati }, { Peruzzi, Strozzi }, { Ridolfi, Strozzi }, { Ridolfi, Tornabuoni } }

(a) Textual Presentation

	Acciaiuoli	Albizzi	Barbadori	Bischeri	Castellani	Ginori	Guadagni	Lamberteschi	Medici	Pazzi	Peruzzi	Pucci	Ridolfi	Salviati	Strozzi	Tornabuoni
Acciaiuoli	.	.	.	.	.	.	.	.	1	.	.	.	.	.	.	.
Albizzi	.	.	.	.	.	1	1	.	1	.	.	.	.	.	.	.
Barbadori	.	.	.	.	1	.	.	.	1	.	.	.	.	.	.	.
Bischeri	.	.	.	.	.	.	1	.	.	.	1	.	.	.	1	.
Castellani	.	.	1	.	.	.	.	.	.	.	1	.	.	.	1	.
Ginori	.	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Guadagni	.	1	.	1	.	.	.	1	.	.	.	.	.	.	.	1
Lamberteschi	.	.	.	.	.	.	1	.	.	.	.	.	.	.	.	.
Medici	1	1	1	.	.	.	.	.	.	.	.	.	1	1	.	1
Pazzi	.	.	.	.	.	.	.	.	.	.	.	.	.	1	.	.
Peruzzi	.	.	.	1	1	.	.	.	.	.	.	.	.	.	1	.
Pucci	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Ridolfi	.	.	.	.	.	.	.	.	1	.	.	.	.	.	1	1
Salviati	.	.	.	.	.	.	.	.	1	1	.	.	.	.	.	.
Strozzi	.	.	.	1	1	.	.	.	.	.	1	.	1	.	.	.
Tornabuoni	.	.	.	.	.	.	1	.	1	.	.	.	1	.	.	.

(b) Tabular Presentation



(c) Graphical Presentation

Figure 2.1: Basic forms of graph presentation. Data from Padgett and Ansell (1993), who explain the rise of the Medici family from its marriage (depicted) and business ties

semantic substance of graphs in one particular domain, making the reason for this distinction more obvious.

Design. Unlike the way it is commonly understood, design does not focus on aesthetics, beauty, or elegance. As designer Conran (1996) puts it, design incorporates 98% function and 2% aesthetics. Visualization design is the specification of how substance is mapped to graphical elements. The information visualization perspective on graph drawing implies that the most important aspect of choosing a specification is the effective communication of substance rather than a beautiful and impressive picture. Aesthetics may play a role in speeding up perception, though. We call the ease of reading the *ergonomics* of a visualization. The *effectiveness* of a design depends on how well the substance is recognized from a visualization.

When Tufte (1983, p. 191) states that "design is a choice" he essentially points to the facts that graphical presentations of the same data can look very different, and that their quality can vary significantly. Consequently, visualization should not merely be seen as an instrument to decorate numbers.

Experimental studies recently validated the effect of different properties of graph visualizations on the understanding of relational structures in general (Purchase *et al.*, 1997; Purchase, 1997), and of domain specific substance (Blythe *et al.*, 1996; McGrath *et al.*, 1998). However, there is little work on how to assess the relative effectiveness of specific visualization techniques. We therefore choose an example from the literature on categorical data presentation to illustrate how the quality of visualizations can differ beyond technical brilliance (see Tufte, 1983, pp. 66ff). The example in Figure 2.2 clearly shows how the same data can be presented in very different ways. Apart from the fact that the second graph yields a much calmer view, leaving behind the distortion in the "chartjunk" (Tufte, 1983) of the first graph, it also does not generate the false impression of a substantial and continuous increase in spending. As Tufte convincingly shows, the first graph deploys several visual and statistical tricks to exaggerate the budget, which does not really increase when put in relation to population size.

The question why one form of graphical presentation of categorical data is more effective than another has been analyzed in great detail by Tufte (1983, 1990, 1997) and a number of other authors (*e.g.* Müller, 1991; Wainer, 1997). Tufte (1983, p. 51) lists the following principles of graphical excellence:

- Graphical excellence is the well-designed presentation of interesting data—a matter of substance, of statistics, and of design.

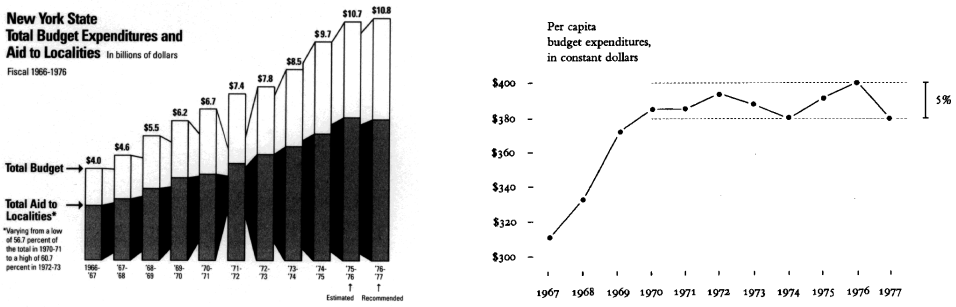


Figure 2.2: From chartjunk to graphical excellence (Tufte, 1983, p. 66/68)

- Graphical excellence consists of complex ideas communicated with clarity, precision, and efficiency.
- Graphical excellence is what gives to the viewer the greatest number of ideas in the shortest time with the least ink in the smallest space.
- Graphical excellence is nearly always multivariate.
- And graphical excellence requires telling the truth about the data.

Algorithm. The procedures used to realize a design specification for the substance of a given graph constitute an aspect that may be equally important. Not only because of runtime considerations, but also because perfect satisfaction of the requirements manifested in a design may not be possible. For instance, a design might require all edges to be represented by straight-line segments of equal length. This clearly is impossible for many graphs. Any algorithm therefore necessarily introduces artifacts or misleading arrangements, even when it gives the best possible solution (with respect to some deviation measure). Moreover, existing approaches to graph drawing often use an algorithm that does not implement a specified design, but implicitly specifies one. It is hence important to be aware of the algorithm, and its peculiarities, underlying a visualization process.

2.1.3 Graphical Design

As analyzed in the pioneering work of graphic designer Bertin (1983), a graphical presentation is composed of a number of topological primitives called *graphical features*. Each graphical feature has a number of properties,

graphical features	graphical variables	
point	*positional*	x-, y-, z-coordinate
curve	*retinal*	size, shape, orientation,
area		brightness, color, transparency,
volume		texture
	temporal	movement and other dynamics

Figure 2.3: Primitives of graphical presentations and their properties (Bertin, 1983, updated)

called *graphical variables*, that may be fixed in advance or varied according to the data. Figure 2.3 lists graphical features for presentations in up to three dimensions together with graphical variables relevant on standard media.

Since the design of a graphical presentation specifies how substance is mapped to graphical variables, it can be subdivided into three major components:

- A *representation* prescribing what kind of feature is to represent which item of a graph and, possibly, constraining selected graphical variables. Graphs, and sometimes only certain classes of graphs, can be represented in various ways. Figure 2.4 gives some of the more common examples. Throughout this thesis, we only consider representations mapping vertices to points and edges to curves.

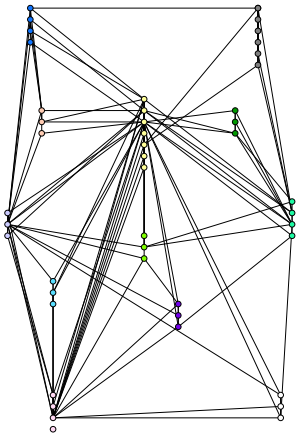
- An effective and ergonomic *layout*, *i.e.* a specification of constraints and criteria for suitable values of graphical variables that determine topological and geometric properties of the presentation.

- An effective and ergonomic *rendering*, *i.e.* a specification of suitable values for graphical variables that are not yet fixed by representation or layout.

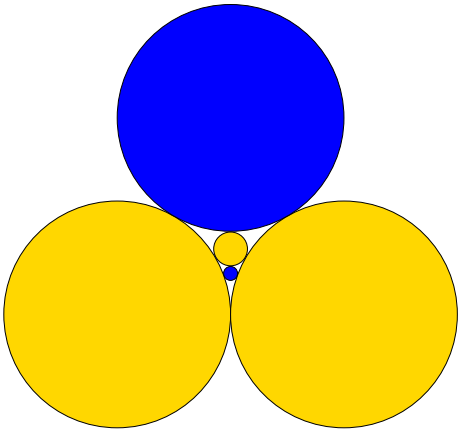
Using experimental evidence, Cleveland and McGill (1984) rank graphical variables according to their effectiveness and accuracy in coding information. Mackinlay (1986) extends and details this list, though no further experiments are carried out to validate these extensions. It seems of no doubt that positional properties, and hence layout considerations, are of utmost importance.

2.2 Layout

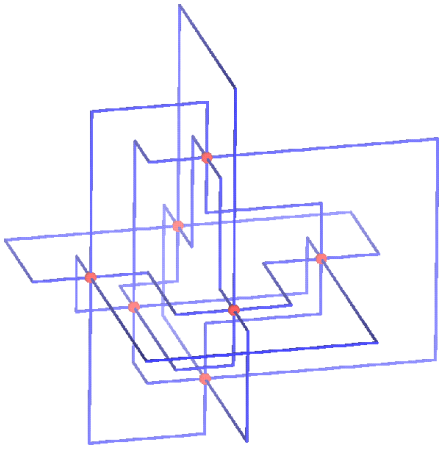
To uniformly describe different graph layout strategies, we develop a fairly general formalism to articulate them in a precise, yet flexible and convenient,



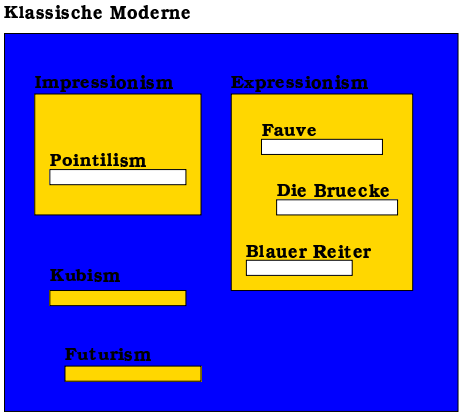
(a) 2D straight-line: Vertices are represented by points in the plane, edges by straight lines. Winning entry for Graph B of the 1997 Graph Drawing Contest (Eades *et al.*, 1997), submitted by Batagelj and Mrvar



(b) Coin graph: Vertices are represented by discs, edges implicitly by touching of discs. K_5 with one edge removed. Redrawn from Brightwell and Scheinerman (1993)



(c) 3D orthogonal: Vertices are represented by points in space, edges by sequences of axis-parallel straight-line segments. K_7 with two bends per edge. Layout by Wood (1996), viewpoint by Webber (1998)



(d) 2D inclusion: Vertices are represented by areas, edges implicitly by area inclusion. Redrawn from Kamps *et al.* (1996)

Figure 2.4: Graph representations. Note that coin graph and 2D inclusion representation are feasible only for restricted classes of graphs

manner. Essentially, graph layout is modeled in terms of constraint optimization. Other formalisms for graph layout are described in Mackinlay (1986), Ding and Mateti (1990), Marks (1991), and Bertolazzi *et al.* (1995).

2.2.1 Layout Models

According to the observations made above, graph layout is the specification and realization of topological and geometric properties of graphical presentations of graphs. The properties to be determined are represented in a set of variables. Since a design might restrict the configuration space, constraints may be imposed on feasible combinations of their values. An objective function measures how well a layout meets design criteria.

Layout Elements. Given a graph and a prescribed representation, each topological or geometric property that is to be determined for some graphical feature is a *layout element*. The layout elements of a straight-line representation, for instance, are simply the positional variables of point features representing vertices. In this case, vertices and layout elements can be identified.

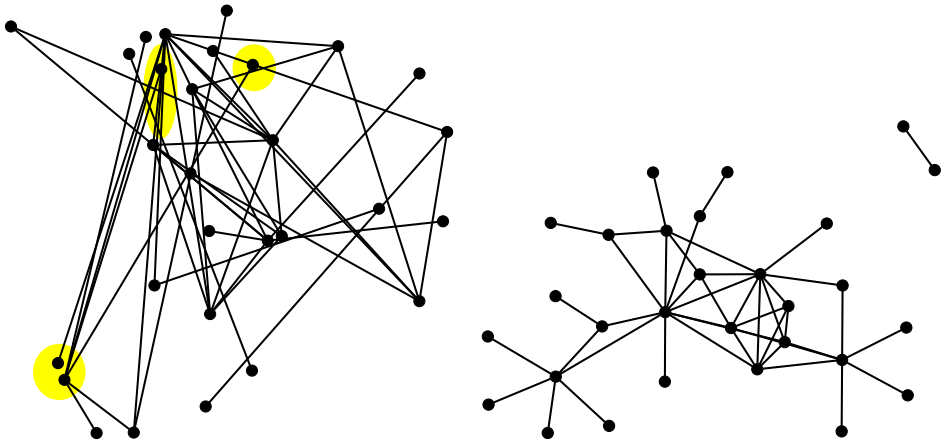
In general there is a set $L = \{\lambda_1, \ldots, \lambda_s\}$ of layout elements, and for each $\lambda \in L$ let there be a set $\mathcal{X}_\lambda$ of feasible values of the respective variable. Then, every vector $x \in \mathcal{X} = \mathcal{X}_{\lambda_1} \times \cdots \times \mathcal{X}_{\lambda_s}$ is called a *layout*. We write $x_\Lambda = (x_\lambda)_{\lambda \in \Lambda}$ for the partial layout of all layout elements in $\Lambda \subseteq L$.

Constraints. Certain values for one layout element may prohibit otherwise feasible values of other layout elements. In Figure 2.4(a), clustered vertices are constrained to have the same x-coordinate. To capture such restrictions, let $R_{\{\lambda_{i_1},\ldots,\lambda_{i_k}\}} \subseteq \mathcal{X}_{\lambda_{i_1}} \times \cdots \times \mathcal{X}_{\lambda_{i_k}}$, $1 \leq i_1 < \ldots < i_k \leq s$, be a k-ary constraint, $\mathcal{R}_k = \bigcup_{1 \leq i_1 < \ldots < i_k \leq s} R_{\{\lambda_{i_1},\ldots,\lambda_{i_k}\}}$ be the set of all k-ary constraints, and $\mathcal{R} = \bigcup_{i=1}^{k} \mathcal{R}_k$ the set of all constraints. Then,

$$\mathcal{X} \cap \mathcal{R} \stackrel{\text{def}}{=} \{x \in \mathcal{X} : x_\Lambda \in R_\Lambda \text{ for all } R_\Lambda \in \mathcal{R}, \Lambda \subseteq L\}$$

is the set of all *feasible layouts*. For convenience we also use $\mathcal{X} \cap R$, or $\mathcal{X} \cap \mathcal{R}'$, to denote those layouts fulfilling a specific constraint $R \in \mathcal{R}$, or a set of constraints $\mathcal{R}' \subseteq \mathcal{R}$, respectively.

Objective functions. It was already indicated that some layouts are more effective than others in conveying the substance of a graph. Likewise, some graph layouts are more ergonomic than others. To assess the effectiveness and ergonomics of a layout x, an objective function $U : \mathcal{X} \cap \mathcal{R} \to \mathbb{R}$ measures



(a) random layout (b) ergonomic layout

Figure 2.5: Different layouts of an abstract graph

conformance with design criteria. For convenience this objective function is usually defined for infeasible layouts as well, *i.e.* $U : \mathcal{X} \to \mathbb{R}$. Unfortunately, it seems rather difficult to devise such global objective function assessing the effectiveness of a given layout directly.

Figure 2.5 shows two different layouts of the same abstract graph. Shaded areas indicate ergonomic problems like vertices that are too close to each other, edges that are too long, vertices that are too close to edges, or pairs of edges forming narrow angles. These undesirable configurations are local in the sense that they involve only few layout elements. We hence base objective functions on design criteria for configurations of (mostly) small subsets of layout elements. Each criterion evaluates, how bad a particular local configuration is. Therefore, the objective function is defined as a cost function combining local criteria. It is to be minimized over $\mathcal{X} \cap \mathcal{R}$ to obtain an effective and ergonomic layout.

The interaction of layout elements jointly appearing in layout criteria is modeled by an *interaction graph* $\mathcal{G} = (L, \mathcal{E})$ obtained from a *neighborhood system* $\eta = \bigcup_{\lambda \in L} \eta_\lambda$, where the *neighborhood* $\eta_\lambda \subseteq L \setminus \{\lambda\}$ is the set of layout elements for which the value assigned to λ is relevant in terms of layout quality. Since these interactions are assumed to be symmetric, we have $\lambda_2 \in \eta_{\lambda_1} \Leftrightarrow \lambda_1 \in \eta_{\lambda_2}$ for all $\lambda_1, \lambda_2 \in L$, so $\mathcal{G}$ is undirected. Let $\mathcal{C} = \mathcal{C}(\eta)$be a set of *cliques* in $\mathcal{G}$, *i.e.* the set of all subsets $C \subseteq L$ with $\{\lambda_1, \lambda_2\} \in \mathcal{E}$ for all pairs $\lambda_1, \lambda_2 \in C$. The badness of a configuration on $C \in \mathcal{C}$ is expressed using an *interaction potential* (or *potential* for short), which may be any function

$U_C : \mathcal{X} \to \mathbb{R}$ for which

$$x_C = y_C \quad \Rightarrow \quad U_C(x) = U_C(y)$$

holds for all $x, y \in \mathcal{X}$. Our objective function is the linear combination of all potentials, $U(x) = \sum_{C \in \mathcal{C}} U_C(x)$. While U_C is sometimes called a *local criterion*, $U(x)$ is called the *energy* of layout x.

We call $U : \mathcal{X} \cap \mathcal{R} \to \mathbb{R}$ the *layout model*, because it summarizes the part of a graphical design that specifies layout elements, constraints on the combination of their values, and criteria for good layout. The energy is easily recognized to induce a Gibbs distribution which assigns probabilities

$$P(X = x) = \frac{1}{Z} e^{-U(x)}$$

where $Z = \sum_{y \in \mathcal{X}} e^{-U(y)}$ is a normalizing constant, to layouts $x \in \mathcal{X}$. Random variable X is hence a Gibbs random field. For simplicity we call U, X, and its distribution P^X a *random field layout model* for G. Probabilities $P(X = x)$ depend on the energy only, with a layout of low energy being more likely than a layout of high energy. Since Gibbs distributions are at the heart of thermodynamics and statistical mechanics, they are well studied in many respects (Guyon, 1995; Li, 1995; Winkler, 1995). A primer on the utilization of random field theory is the development of a framework for dynamic layout models in Section 2.2.2.

Two layout approaches are usually distinguished in the literature on graph drawing: *declarative* and *algorithmic* (Cruz and Tamassia, 1994). In declarative graph layout, users are typically allowed to specify a set of constraints like spatial clustering, alignment, etc. The system then tries to satisfy these constraints either completely or as many as possible. Due to the inherent complexity of the resulting constraint satisfaction problems, these systems are usually slow. In algorithmic graph layout, the emphasis is on criteria of layout quality defining the objective function. Minimization of a suitable objective function is more efficient than constraint solving in general. Moreover, the focus on algorithms often allows to decompose a strategy into steps that are shared by other strategies. Like in most of the graph drawing literature, the focus of this thesis is on algorithmic approaches to graph layout.

2.2.2 Dynamic Layout Models

Up to now graphs were considered static objects. We here develop a generic approach to layout of a *dynamic graph*, *i.e.* a sequence of graphs arising from

repeated modification of an initial graph. These modifications may be due to user interaction, algorithms, or other underlying processes determining the graph.

Looking at graphical presentations of graphs, users develop a *mental map* of the diagram (Eades *et al.*, 1991). When a diagram changes, this mental map should be preserved as closely as possible to enable users to easily regain familiarity with the presentation. Until now, there are only few approaches to dynamic graph layout, most of them tied to specific classes of admissible graphs or representations (Böhringer and Paulisch, 1990; Cohen *et al.*, 1992, 1995; North, 1996a). We essentially argue that dynamic graph layout can be regarded a special case of static layout, where an additional criterion captures the notion of stability between consecutive layouts.

A general analysis of the dynamic layout problem is given in North (1996a). Dynamic layout is considered to consist of a sequence of static layout problems subject to *consistency* (retain important properties), *stability* (allow only moderate changes), and *readability* (good static design) demands. Furthermore, the *logical update* from one layout to the next may be extended into a sequence of *physical updates* to ease the transition. This method of gradual transformation will be used in our case study of a dynamic graph in Section 3.3. The purpose of this section is to provide a seamless integration of dynamic layout in our general layout formalism.

Let there be a (finite or infinite) sequence of graphs $G(1), G(2), \ldots$, for which (static) layout models $X(1), X(2), \ldots$ are given. The assumption of an infinite sequence is appropriate when the graph is modified unpredictably. Such situations occur in user interaction, network control, phone-call recording, and so on. In these cases, neither the next graph, nor the number of graphs to come is known at any point in time.

In analogy to the above definition of a random field layout model, we now want to construct a random vector $(X(1), X(2), \ldots)$ that describes the relative effectiveness of all sequences of layouts, such that optimal sequences correspond to modes of the joint distribution $P^{(X(1),X(2),\ldots)}$. Before we go into the details of how to use this approach to devise suitable models, let's first get a feel for what this means. In the case of independent layout models, we have

$$P\big(X(1) = x(1), X(2) = x(2), \ldots\big) = \prod_i P\big(X(i) = x(i)\big)$$

Independent random fields correspond to a strategy that layouts each individual graph according to its own model and does not care about the user's mental map (except for, possibly, in the physical update). An optimal sequence is hence computed by individually maximizing all $P(X(i) = x(i))$

over $x(i) \in \mathcal{X}(i)$. Since gradual transitions between consecutive layouts are intended, dependencies among the individual models must be introduced.

The joint probability of a finite sequence $G(1), \ldots, G(t)$ can be rewritten into conditional (transition) probabilities

$$P\big(X(1) = x(1), \ldots, X(t) = x(t)\big) = \prod_{i=1}^{t} P\big(X(i) = x(i) \,\big|\, X(< i) = x(< i)\big)$$

where $X(< i) = x(< i)$ is shorthand for $X(1) = x(1), \ldots, X(i-1) = x(i-1)$. However, knowledge of all graphs and dependencies is required to obtain any $x(i)$ of a sequence $x(1), \ldots, x(t)$ that maximizes the expression. Such knowledge is typically not provided.[2] It seems therefore reasonable to restrict our attention to the case that each layout of the sequence (finite or infinite) has to be computed before anything about the next graph is known.

With no look-ahead available, we obtain the following formalization of the dynamic layout problem. At time $t > 1$, we are given graphs $G(1), \ldots, G(t)$, static layout models $X(1), \ldots, X(t)$, and layouts $x(1), \ldots, x(t-1)$. The goal is to compute a layout $x(t) \in \mathcal{X}(t)$ that forms a compromise between stability and readability. Obviously, the conditional probability $P(X(t) = x(t) \,|\, X(< t) = x(< t))$ must reflect this notion of compromise. It is hence called the *dynamic layout model* of $G(t)$, and because of the following considerations it is reasonable to maximize this probability.

Suppose, $\mathsf{loss} : \mathcal{X}(t) \times \mathcal{X}(t) \to \{0, 1\}$ is the (imaginary) zero-one loss function of choosing $x(t)$, when the best choice is x,

$$\mathsf{loss}(x(t), x) = \begin{cases} 0 & \text{if } x(t) = x \\ 1 & \text{if } x(t) \neq x \end{cases}$$

With this loss function, the risk $\sum_{x \in \mathcal{X}(t)} \mathsf{loss}(x(t), x) \cdot P(X(t) = x \,|\, X(< t) = x(< t)) = 1 - P(X(t) = x(t) \,|\, X(< t) = x(< t))$ of selecting $x(t)$ equals the average probability of error. It is minimized by choosing an $x(t)$ for which $P(X(t) = x(t) \,|\, X(< t) = x(< t))$ is maximized. Observe that other measures of loss yield other decision rules.

In the remainder of this section, we propose a Bayesian approach for specifying dynamic layout models, which basically provides a formalization of common sense. Note that, by Bayes' rule,

$$\max_{x(t) \in \mathcal{X}(t)} P\big(X(t) = x(t) \,\big|\, X(< t) = x(< t)\big)$$
$$= \max_{x(t) \in \mathcal{X}(t)} \frac{P\big(X(< t) = x(< t) \,\big|\, X(t) = x(t)\big) \cdot P\big(X(t) = x(t)\big)}{P\big(X(< t) = x(< t)\big)}$$

[2] Observe that there are applications, like animation, where the complete sequence might indeed be known in advance.

which is proportional to

$$\max_{x(t)\in\mathcal{X}(t)} P\big(X(<t)=x(<t)\,\big|\,X(t)=x(t)\big)\cdot P\big(X(t)=x(t)\big)$$

The term $P(X(t)=x(t))$ is of course the static layout model for $G(t)$. It therefore reflects the notions of consistency and readability formalized in $X(t)$. $P(X(<t)=x(<t)\,|\,X(t)=x(t))$ is the likelihood of the sequence $x(1),\ldots,x(t-1)$ to lead to an assumed layout $x(t)$ of $G(t)$. We have thus found a place to incorporate the notion of stability, which is why $P(X(<t)=x(<t)\,|\,X(t)=x(t))$ is called the *stability model*.

In summary, we have argued that a dynamic layout model for a graph $G(t)$ can be composed using its static layout model $P^{X(t)}$, and a measure $P^{(X(<t)|X(t))}$ of change with respect to layouts $x(1),\ldots,x(t-1)$ of preceding graphs. If this measure is formulated in terms of a Gibbs distribution with energy function $U^{(X(<t)|X(t))}$, the dynamic layout model is also a Gibbs distribution with energy function

$$U^{(X(t)|X(<t))}\big(x(t)\,\big|\,x(<t)\big)=U^{(X(<t)|X(t))}\big(x(<t);x(t)\big)+U^{X(t)}(x(t)\big)$$

which leads to the conclusion that it is reasonable to incorporate stability as just another criterion in an objective function.

We have thus developed an abstract formulation of a general principle for dynamic graph layout that complements given static models. Note that in the particular case of straight-line embeddings Lüders *et al.* (1995) arrived at the idea of incorporating stability expressions in the layout objective function on an ad-hoc basis. Examples for the utilization of stability criteria in dynamic layout models are given in Sections 3.3 and 5.3.

2.2.3 Layout Computation

Graph layout was formalized as an optimization problem in the previous sections. Effective and ergonomic layouts are obtained by minimizing an energy function $U(x)=\sum_{C\in\mathcal{C}}U_C(x)$ over all $x\in\mathcal{X}\cap\mathcal{R}$. Due to the universality of the problem definition, energy minimization is easily seen to be $\mathcal{NP}$-hard in general. We here give a simple proof showing that the problem remains $\mathcal{NP}$-hard even for very restricted models.

Theorem 2.1 *Energy minimization for random field layout models defined on graphs $G=(V,E)$ is $\mathcal{NP}$-hard, even if $L=V$, $\mathcal{G}=G$, $\mathcal{C}=E$, $\mathcal{X}=\{0,1\}^L$, and U contains at most quadratic terms.*

■ **Proof** The corresponding decision version is obviously in $\mathcal{NP}$. To show that it is $\mathcal{NP}$-complete, we transform an arbitrary instance $G = (V, E)$ of SIMPLE MAX CUT (unit weight variant of problem [ND16] in Garey and Johnson 1991) into a random field layout model X of G, such that there is a layout of energy $|E| - k$, if and only if there is a cut with k edges. Using interaction potentials

$$U_{\{u,v\}}(x) = x_u \cdot x_v + (1 - x_u) \cdot (1 - x_v) = \begin{cases} 0 & \text{if } x_u \neq x_v \\ 1 & \text{if } x_u = x_v \end{cases}$$

for all edges $\{u, v\} \in E$, a layout $x \in \{0, 1\}^V$ is assigned an energy equal to the number of edges that have the same value assigned to both endpoints. □

The prototyping approach taken in Chapter 3 usually leads to $\mathcal{NP}$-hard models. In the remainder of this section we therefore review optimization techniques for approximate minimization of arbitrary energy functions. A number of interesting special cases do allow for efficient optimization, though. Examples of such cases are the subject of Chapters 4 and 5, while many others are listed in the graph drawing bibliography (Di Battista *et al.*, 1994) and in Di Battista *et al.* (1999).

Locality. Local criteria for good graph layout are usually conflicting. This and the fact that we are concerned with graphical presentation rather than technical applications of graph layout indicate that optimality of a layout might not be a prerequisite. In many cases, locally optimal solutions will be satisfactory. They can be obtained by local search methods that iteratively improve an initial layout by changing the value of, typically, only a single layout element. The following observation is crucial for most iterative methods.

Lemma 2.2 *For every random field layout model X,*

$$\begin{aligned} P\big(X_\lambda = x_\lambda \,\big|\, X_{L-\lambda} = x_{L-\lambda}\big) &= P\big(X_\lambda = x_\lambda \,\big|\, X_{\eta_\lambda} = x_{\eta_\lambda}\big) \\ &= \frac{1}{Z_\lambda}\, e^{-\sum_{C \in \mathcal{C} \,:\, \lambda \in C} U_C(x)} \end{aligned}$$

where $Z_\lambda = \sum_{x_\lambda \in \mathcal{X}_\lambda} e^{-\sum_{C \in \mathcal{C} \,:\, \lambda \in C} U_C(x)}$.

■ **Proof**

$$\begin{aligned} P(X_\lambda = x_\lambda \,|\, X_{L-\lambda} = x_{L-\lambda}) &= \frac{P(X = x)}{P(X_{L-\lambda} = x_{L-\lambda})} \\ &= \frac{Z^{-1} \cdot \exp\{-U(x)\}}{\sum_{x_\lambda} Z^{-1} \cdot \exp\{-U(x)\}} = \frac{\exp\big\{-\sum_{C \in \mathcal{C}} U_C(x)\big\}}{\sum_{x_\lambda} \exp\big\{-\sum_{C \in \mathcal{C}} U_C(x)\big\}} \end{aligned}$$

and, since $U_C(x)$ is independent of x_λ if $\lambda \notin C$,

$$= \frac{\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\notin C} U_C(x)\right\}\cdot\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\in C} U_C(x)\right\}}{\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\notin C} U_C(x)\right\}\cdot\sum_{x_\lambda}\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\in C} U_C(x)\right\}}$$
$$= \frac{\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\in C} U_C(x)\right\}}{\sum_{x_\lambda}\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\in C} U_C(x)\right\}}$$

which is the second equality, and finally

$$= \frac{\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\in C} U_C(x)\right\}}{\sum_{x_\lambda}\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\in C} U_C(x)\right\}} \cdot \frac{\sum_{x_{\lambda'}\,:\,\lambda'\notin\eta_\lambda\cup\{\lambda\}}\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\notin C} U_C(x)\right\}}{\sum_{x_{\lambda'}\,:\,\lambda'\notin\eta_\lambda\cup\{\lambda\}}\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\notin C} U_C(x)\right\}}$$
$$= \frac{\sum_{x_{\lambda'}\,:\,\lambda'\notin\eta_\lambda\cup\{\lambda\}}\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\in C} U_C(x)\right\}}{\sum_{x_{\lambda'}\,:\,\lambda'\notin\eta_\lambda}\exp\left\{-\sum_{C\in\mathcal{C}\,:\,\lambda\in C} U_C(x)\right\}}$$
$$= \frac{P(X_{\eta_\lambda\cup\{\lambda\}} = x_{\eta_\lambda\cup\{\lambda\}})}{P(X_{\eta_\lambda} = x_{\eta_\lambda})} = P\left(X_\lambda = x_\lambda \mid X_{\eta_\lambda} = x_{\eta_\lambda}\right)$$

□

The probabilities appearing in the above lemma are called the *local characteristics* of the random field.[3] They describe the benefit of assigning x_λ to layout element λ, when the value of every other layout element $\lambda' \in L - \lambda$ is fixed to be $x_{\lambda'}$. Lemma 2.2 states that these probabilities depend only on the neighbors of λ. The best value of x_λ, given all the others, is obtained by maximizing this probability or, equivalently, minimizing the *local energy* $\sum_{C\in\mathcal{C}:\lambda\in C} U_C(x)$ over $\mathcal{X}_\lambda$.

By repeatedly minimizing the local energy of a layout model for each $\lambda \in L$, a local minimum of its (unconstrained) energy is obtained. This method is introduced in Besag (1986), where it is called *iterated conditional modes (ICM)*. It is known to display rapid convergence to a local minimum of the global energy. Several graph layout algorithms are closely related to this method (Eades, 1984; Kamada and Kawai, 1989; Sugiyama and Misue, 1995), but require differentiable energy functions. The following method does not impose restrictions on the class of interaction potentials.

[3] If $P(X_\lambda = x_\lambda |\, X_{L-\lambda} = x_{L-\lambda}) = P(X_\lambda = x_\lambda |\, X_{\eta_\lambda} = x_{\eta_\lambda})$ holds for all $\lambda \in L$, the random field is said to satisfy the *Markov property*. If a strictly positive random field is defined in terms of consistent local characteristics, it is called a *Markov random field*. An important result from stochastic theory states that Markov and Gibbs random fields are equivalent with respect to the same interaction graph (a detailed proof is contained in, *e.g.*, Griffeath, 1976).

Annealing. A common way of avoiding poor local minima is the use of annealing-type algorithms. *Simulated annealing* is a general optimization method for large-scale combinatorial problems, first introduced in Metropolis *et al.* (1953) and Kirckpatrick *et al.* (1983). Elaborate treatments are given by Aarts and Korst (1989) and van Laarhoven and Aarts (1988). Roughly speaking, the algorithm iteratively modifies a candidate layout. In each iteration, the modified layout is either accepted to be the new candidate layout, or it is rejected. It is accepted, if its energy is smaller than the energy of the current candidate. In order to avoid poor local minima, layouts with higher energy are accepted with probability $e^{-\Delta U/T}$, where ΔU is the energy difference and $T > 0$ the parameter of the annealing, the *temperature*.[4] Convergence is enforced by slowly lowering T according to some *cooling schedule*.

We therefore parameterize a random field with the real valued temperature $T > 0$. The joint distribution becomes

$$P\left(X(T) = x\right) = \frac{1}{Z(T)}\, e^{-U(x)/T}$$

where $Z(T) = \sum_{y \in \mathcal{X}} \exp\left\{-U(y)/T\right\}$. The temperature parameter controls the effect of the energy function on the joint distribution. For higher values of T, $P^{X(T)}$ is close to a uniform distribution over all layouts, while the distribution peaks more sharply for smaller values of T. In a temperatured random field $X(T)$, the local characteristics become

$$P\left(X_\lambda(T) = x_\lambda \,|\, X_{\eta_\lambda}(T) = x_{\eta_\lambda}\right) = \frac{1}{Z_\lambda(T)} \exp\left\{ - \sum_{C \in \mathcal{C}: \lambda \in C} U_C(x)/T \right\}$$

where $Z_\lambda(T) = \sum_{x_\lambda \in \mathcal{X}_\lambda} \exp\left\{-U(x)/T\right\}$. The so-called *Gibbs sampler* (Geman and Geman, 1984) approximates $P^{X(T)}$ by repeatedly sampling from these temperatured local characteristics. Since $\lim_{T \downarrow 0} P^{X(T)}$ is the uniform distribution over all layouts of minimum energy, T is decreased over time to obtain a mode of the original distribution.

Without going into too much detail, we can gain some insight about the relation to simulated annealing by considering two layouts x and y that differ only in the value assigned to one layout element λ, *i.e.* a typical pair of candidate and modified candidate layout. The ratio of their local characteristics

[4] Note that "temperature" is sometimes also used to denote an adaptive parameter controlling the displacement in gradient search approaches like the one of Fruchterman and Reingold (1991).

with respect to λ is

$$\begin{aligned}\frac{P\left(X_\lambda(T)=y_\lambda \mid\ X_{\eta_v}(T)=y_{\eta_\lambda}\right)}{P\left(X_\lambda(T)=x_\lambda \mid\ X_{\eta_v}(T)=x_{\eta_\lambda}\right)} &= \frac{Z_v^{-1}(T)\cdot\exp\left\{-\sum_{C\in\mathcal{C}:\lambda\in C}U_C(y)/T\right\}}{Z_v^{-1}(T)\cdot\exp\left\{-\sum_{C\in\mathcal{C}:\lambda\in C}U_C(x)/T\right\}} \\ &= \exp\left\{-\sum_{C\in\mathcal{C}:\lambda\in C}\left(U_C(y)-U_C(x)\right)/T\right\}\end{aligned}$$

since $y_{\lambda'}=x_{\lambda'}$ for all $\lambda'\in L-\lambda$. Clearly,

$$\min\left\{1\,,\,\exp\left\{-\sum_{C\in\mathcal{C}:\lambda\in C}\left(U_C(y)-U_C(x)\right)/T\right\}\right\}$$

is the acceptance probability of simulated annealing, provided that a new candidate solution is generated by modifying a single layout element λ. During the algorithm, the energy $U(x)$ need never be computed entirely, since the transition probabilities depend on the energy differences only. The role of the interaction graph $\mathcal{G}$ is now evident, because an implementation needs to compute energy differences only on cliques of $\mathcal{G}$ that have non-zero interaction potentials and contain the respective layout element. Since the above procedure differs from the Gibbs sampler only in the sampling strategy used to approximate $P^{X(T)}$, it is often termed *Metropolis sampler*. A comparative study of both algorithms is Chen and Schmeiser (1993).

Simulated annealing is used for graph layout, for instance, in Davidson and Harel (1996) and Cruz and Twarog (1996), even though it is often disregarded because of its computational cost (Brandenburg *et al.*, 1996). However, faster algorithms based on gradients (Eades, 1984; Kamada and Kawai, 1989; Fruchterman and Reingold, 1991; Frick *et al.*, 1995; Sugiyama and Misue, 1995) require energy functions with continuous partial derivatives. It is interesting to note that they directly correspond to the limit case $T=0$ of the annealing algorithms and are hence implementations of ICM, *i.e.* they are annealing algorithms using a cooling schedule that immediately freezes the system. Finally notice that quite a number of graph layout algorithms are implemented in a batch-like fashion (*e.g.* Eades 1984; Kamada and Kawai 1989; Sugiyama and Misue 1995). A new value is computed independently for each layout element. Afterwards, all layout elements are updated simultaneously. This corresponds directly to a parallel algorithm with synchronous, independent updates of all layout elements. In general, such algorithms converge to a different limit distribution, and oszillation effects may occur.

It is important to note that unary constraints cause no difficulties for general optimization techniques based on annealing. On the other hand, k-ary constraints with $k > 1$ in general disconnect the search space. The introduction of an indicator variable

$$R(x) = \begin{cases} 0 & \text{if } x \in \mathcal{X} \cap \mathcal{R} \\ 1 & \text{otherwise} \end{cases}$$

enables us to gradually increase a penalty for infeasible layouts by replacing the energy with

$$U(x) + \rho(T) \cdot R(x)$$

where $\rho(T)$ goes to infinity for $T \downarrow 0$. While Geman *et al.* (1990) proof that the annealing process converges in probability to an optimal solution (with suitably chosen cooling schedules and penalty functions), this method cannot be expected to be efficient enough to be used in a general optimizer for layout models.

Thorough treatments of general energy minimization methods are given in Azencott (1990), Guyon (1995), Winkler (1995), and Li (1995). See also Pelillo (1997).

Implemented algorithms exist for a huge number of layout models. Many are available in more general systems like GRAPHLET (Himsolt, 1996), the GRAPH DRAWING SERVER (Bridgeman *et al.*, 1996), and GRAVIS (Lauer *et al.*, 1997), or in libraries like GDTOOLKIT[5] and AGD (Mutzel *et al.*, 1998). A drawback of most implementations is that they usually allow but a few parameters of a layout model to be varied. In most cases, only constants for minor design variations like desired edge lengths and bounds on the number of bends an edge may have, or algorithmic parameters like the number of iterations may be specified.

Genericity is a major concern in the design of the AGD library. However, the focus is on generic algorithms rather than generic layout models. Modules implementing steps of popular layout algorithms may be replaced with different implementations, but it is by coincidence that, *e.g.*, the `RankAssignmentModule` determining levels for layered graph layouts (as in Sugiyama *et al.* 1981) also constitutes a major part of the design.

Our graph layout formalization presented in Section 2.2.1 decomposes the problem into several parts: layout elements, neighborhoods, interaction potentials, and constraints. The implied separation lends itself to generic implementation of a flexible layout system, in which these parts are freely

[5]Project home page at `http://www.dia.uniroma3.it/~gdt/`.

combined to form specific models. A prototype version of such a system has been used to generate layouts from the models described in Chapter 3. The implementation consists of a set of fundamental neighborhood types and interaction potentials, to which others can be added. Since our concerns are flexibility and generality, simulated annealing is used for energy minimization. The optimization module can nevertheless be replaced with different implementations. The layout model developed in Section 3.4 provides an example of how the general optimization method can be substituted for a more efficient algorithm exploiting the specific characteristics of the final model.

Chapter 3

General Energy Layouts

Graphs are used as abstractions in numerous fields of application. Consequently, the same structure can mean quite different things depending on the context. To effectively visualize a graph, the specific type of information it represents must govern the graphical design. Drawing traditions and restrictions like available media may further influence the specification of layout models.

The layout framework introduced in the previous chapter separates the design of a graphical presentation from its algorithmic realization. Though inefficient, a universal algorithm can be employed to tackle the resulting optimization problems. Together this enables us to try out different layout models for graphs from a given application without having to implement a new algorithm every time the model is changed. Rapid prototyping of layout models can thus be carried out in cooperation with domain experts evaluating the results. When a satisfactory layout model is found, one can try to improve on computation times by tailoring heuristic methods to this model, or by formulating an equivalent model in a way that is amenable to efficient algorithms.

Three case studies shall demonstrate the practicality of this approach. To get a better feel for the use of local criteria, though, we first take a look at some well known layout methods from the graph drawing literature. Then we provide layout models for graphs representing structures as diverse as social networks, dynamic linkages of a *World Wide Web* site, and time table entries of several European public transport systems by drawing from the toolbox of constraints and local criteria. A number of example layouts is given to indicate the appropriateness of the models thus devised. In one particular case we also demonstrate that computation times can substantially be reduced by subsequent replacement of the general energy minimization routine.

3.1 Force Directed Placement

The famous *spring embedder* (Eades, 1984) is the force behind a whole class of layout algorithms typically subsumed under *force directed placement*.[1] These are layout algorithms for straight-line representation of general undirected graphs. Recall that the layout elements of straight-line representations are the vertices of the graph. A common feature of the underlying layout models is the exploitation of a physical analogy. Vertices and edges of a graph are regarded as physical objects exerting forces on each other, and a configuration of low internal stress is sought.[2] More precisely, a spring embedder for an undirected graph $G = (V, E)$ assumes repelling forces

$$\frac{c_1}{d(x_u, x_v)^2}$$

between points representing non-adjacent vertices $u, v \in V$. While c_1 is a constant, $d(x_u, x_v)$ denotes the Euclidean distance between positions assigned to layout elements u and v. If $\{u, v\} \in E$, a spring is pretended between the points representing u and v. The spring exerts a force of magnitude

$$c_2 \cdot \log \frac{d(x_u, x_v)}{l}$$

where c_2 and l are further constants. Note that the spring does not exert any force if the distance between x_u and x_v is exactly l. The parameter thus represents the ideal length of the spring. An equilibrium configuration is obtained by iteratively computing all forces present and moving points representing vertices along their resulting force vector.

It seems that relaxed configurations of such force systems satisfy a number of straightforward criteria for ergonomic layout simultaneously, *e.g.* uniform edge length, symmetry, and spatial clustering of dense subgraphs. Note that the layout of Figure 2.5(b) was generated from such model. Since these methods are "easy to understand and relatively simple to code" while "the results can be good" (Di Battista *et al.*, 1999, p. 304), they are very popular. The many conceived variants differ in the forces defined and in the algorithms used to find a configuration in equilibrium. A distinction can be made between approaches actually defining forces, and those assigning a potential

[1] A number of authors have independently presented force directed methods (Tutte, 1963; Fisk *et al.*, 1967; Quinn and Breuer, 1979; McDonald and Pedersen, 1991), but none of these works had comparable impact.

[2] As noted in Fruchterman and Reingold (1991), the term "force" is interpreted quite loosely. Functions modeling forces are usually defined to be efficiently computable rather than to match their physical analogies.

energy to each configuration. While the former define a layout model only implicitly, the latter obviously fit directly into the framework of Chapter 2. We therefore use the term *energy based placement* to seclude approaches with an explicit objective function.

3.1.1 Spring Embedder Refinements

Fruchterman and Reingold (1991) replace the spring embedder's repelling and attracting forces between pairs of vertices by repelling forces

$$f_{\mathsf{rep}}(x_u, x_v) = \frac{l^2}{d(x_u, x_v)}$$

between every pair of vertices u and v, and additional attracting forces

$$f_{\mathsf{attr}}(x_u, x_v) = \frac{d(x_u, x_v)^2}{l}$$

only between adjacent vertices. Parameter l describes the optimum length of a single edge, and is set to $c \cdot \sqrt{c_A/n}$, where c_A is the desired layout area, and c is an experimentally chosen constant. These functions are cheaper to evaluate and steeper, thus resulting in faster convergence. The algorithm to find a stable configuration features an adaptive parameter controlling the maximum displacement allowed. Frick *et al.* (1995) develop this idea further by introducing an independent adaptive parameter for each vertex, and some other heuristics to speed up the computation. They also introduce an additional force dragging the vertices to the barycenter of the layout. The approach is modified for three-dimensional straight-line representations in Bruß and Frick (1996).

While most force directed placement approaches are suitable for undirected graphs only, the variant of Sugiyama and Misue (1995) allows (even mixed in) directed edges.[3] In addition to the springs and repulsive forces of the basic spring embedder, magnetic fields introduce new rotative forces

$$f_{\mathsf{rot}}(x_u, x_v) = c_1 \cdot c_B \cdot d(x_u, x_v)^{c_2} \cdot \sphericalangle_B(x_u, x_v)^{c_3}$$

acting on the springs (*i.e.* the edges). According to possible combinations of edge direction, springs are either non-magnetic, uni-directional magnetic, or bi-directional magnetic springs. Here, c_B denotes the strength of the magnetic field, $\sphericalangle_B(x_u, x_v)$ the angle between the field's orientation and the edge vector (if the spring is bi-directional, it is the minimum of both angles), and c_1, c_2, and c_3 are constants. We make use of this idea in Section 3.3 to depict hierarchies.

[3] Tunkelang (1994, Section 4.1.3) has essentially the same idea, but does not elaborate.

The design goals implicit in the above approaches can be expressed in terms of local criteria by using potential functions with appropriate minima. Since intuitive and controllable local criteria meeting most of these goals have already been introduced in some well known energy based placement methods, we refrain from presenting such potential functions. Instead we review briefly the corresponding literature using the notation introduced in Chapter 2.

3.1.2 Energy Based Placement

Kamada and Kawai (1989) formulate an explicit objective function for good layout. It consists of interaction potentials for every pair of vertices measuring the deviation of the Euclidean distance between two vertex positions from a monotone function in the vertices' graph theoretic distance,

$$U^{(\mathsf{KK})}_{\{u,v\}}(x) = \frac{\left(d(x_u, x_v) - l \cdot d_G(u,v)\right)^2}{d_G(u,v)^2}$$

where l is again the desired length of a single edge and $d_G(u,v)$ denotes the length of a shortest path between u and v in G.[4] A local minimum of $U^{(\mathsf{KK})}(x) = \sum_{u \neq v \in V} U^{(\mathsf{KK})}_{\{u,v\}}(x)$ is computed by iteratively applying a two-dimensional Newton-Raphson method to relocate a single vertex.

An immediate problem of the basic spring embedder is the lack of control over distance between a vertex and an edge. Clearly, no edge should pass by a non-incident vertex closer than necessary. While the energy function of Kamada and Kawai (1989) indirectly accounts for this criterion, Davidson and Harel (1996) introduce a corresponding potential

$$U^{(\mathsf{DH})}_{\{u,v,w\}}(x) = \frac{c_\varrho}{d\left(x_v; x_u, x_w\right)^2}$$

where $d(x_v; x_u, x_w) = \min_{t \in [0,1]} d\left(x_\lambda, x_{\lambda_1} + t \cdot (x_{\lambda_2} - x_{\lambda_1})\right)$ is the minimum distance between x_v and any point on the straight-line segment connecting x_u and x_w, $v \notin \{u,w\} \in E$, and c_ϱ is a repulsion factor. This potential is

[4] This is similar to multidimensional scaling. If the input matrix is $(d_G(u,v))_{u,v \in V}$, multidimensional scaling is to minimize

$$\frac{\sum_{u,v \in V} \left(d(x_u, x_v) - d_G(u,v)\right)^2}{\sum_{u,v \in V} d(x_u, x_v)^2}$$

over $x \in \mathbb{R}^{2 \cdot |V|}$ (Kruskal and Wish, 1978). Roughly speaking, the energy of Kamada and Kawai (1989) renders deviation from larger desired distances less important, thus emphasizing short range relations.

interaction potential	interpretation
$\mathsf{Attraction}\,(x_{\lambda_1}, x_{\lambda_2} \mid c_\alpha) = c_\alpha \cdot d(x_{\lambda_1}, x_{\lambda_2})^2$	λ_1 and λ_2 should be as close as possible
$\mathsf{Repulsion}\,(x_{\lambda_1}, x_{\lambda_2} \mid c_\varrho) = \dfrac{c_\varrho}{d(x_{\lambda_1}, x_{\lambda_2})^2}$	λ_1 and λ_2 should be as far apart as possible
$\mathsf{Repulsion}\,(x_\lambda; x_{\lambda_1}, x_{\lambda_2} \mid c_\varrho) = \dfrac{c_\varrho}{d(x_\lambda; x_{\lambda_1}, x_{\lambda_2})}$	λ should be as far from the straight-line segment connecting λ_1 and λ_2 as possible
$\mathsf{Distance}\,(x_{\lambda_1}, x_{\lambda_2} \mid c_\delta) = \dfrac{c_\delta^4}{d(x_{\lambda_1}, x_{\lambda_2})^2}$	λ_1 and λ_2 should be at distance c_δ from each other
$\mathsf{Angle}\,(x_u, x_v, x_w \mid c_\varrho) = c_\varrho \cdot \sphericalangle(x_v; x_u, x_w)$	the angle formed by line segments $\overline{x_v x_u}$ and $\overline{x_v x_w}$ should be as large as possible
$\mathsf{Crossing}\,(x_{\lambda_1}, x_{\lambda_2}; x_{\lambda_3}, x_{\lambda_4}) = \begin{cases} 1 & \text{if } \overline{x_{\lambda_1} x_{\lambda_2}}, \\ & \overline{x_{\lambda_3} x_{\lambda_4}} \text{ cross} \\ 0 & \text{otherwise} \end{cases}$	avoid crossings

Figure 3.1: Fundamental local criteria in straight-line representations

used in combination with attraction and repulsion potentials similar to the forces of Fruchterman and Reingold (1991),

$$U^{\mathsf{DH}}_{\{u,v\}}(x) = \begin{cases} \dfrac{c_\varrho}{d(x_u, x_v)^2} + c_\alpha \cdot d(x_u, x_v)^2 & \text{if } \{u, v\} \in E \\ \dfrac{c_\varrho}{d(x_u, x_v)^2} & \text{otherwise} \end{cases}$$

where the ratio of the repulsion constant c_ϱ and the attraction constant c_α determines the desired edge length $l = \sqrt[4]{c_\varrho / c_\alpha}$. Finally, there are additional single vertex criteria

$$U^{(\mathsf{DH})}_{\{v\}}(x) = \frac{c_\varrho}{d(x_v; \mathsf{top})^2} + \frac{c_\varrho}{d(x_v; \mathsf{bottom})^2} + \frac{c_\varrho}{d(x_v; \mathsf{left})^2} + \frac{c_\varrho}{d(x_v; \mathsf{right})^2}$$

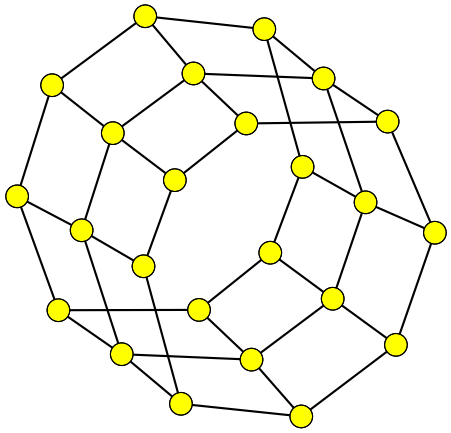
simulating repulsion from the top, bottom, left, and right side of the layout area, and four-vertex potentials (for every pair of disjoint edges) counting the number of edge crossings in the layout. Simulated annealing is used to minimize this complex, discrete energy function. Tunkelang (1994) presents several heuristics to improve on running time, but otherwise uses a subset of the above criteria. For three-dimensional layouts Cruz and Twarog (1996) replace the vertex-edge distance and crossing potential by an edge-edge repulsion potential.

Some important local criteria, many of which will be used in subsequent sections, are summarized in Figure 3.1. Note that most of them already appear in Davidson and Harel (1996). Vertical bars in arguments separate parameters from actual variables. While parameters may sometimes be omitted, replacing some variables by fixed values yields further potentials that can be used to express additional properties. Observe that potentials $U_C^{(i)}$ can be scaled and weighted by functions $f_C^{(i)} : \mathbb{R} \to \mathbb{R}$, so that their combination yields an energy

$$U(x) = \sum_i \sum_{C \in \mathcal{C}} f_C^{(i)} \left(U_C^{(i)}(x) \right)$$

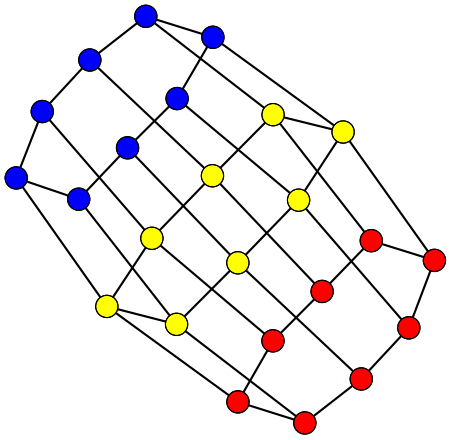
which may render some criteria more important than others. In particular, negative parameters can be used to reverse the objective of a potential.

Many different layout models can be built out of a small number of fundamental local criteria like those in Figure 3.1. The following simple examples are meant to demonstrate how a graph can be twisted and turned by using different potentials. Layout models are described in tables with one row for each criterion, stating the layout elements involved and the potential assigned to the corresponding clique of the interaction graph. The set of layout elements, the interaction graph, *etc.* are implicit. The most basic combination of potentials resembles the original spring embedder.



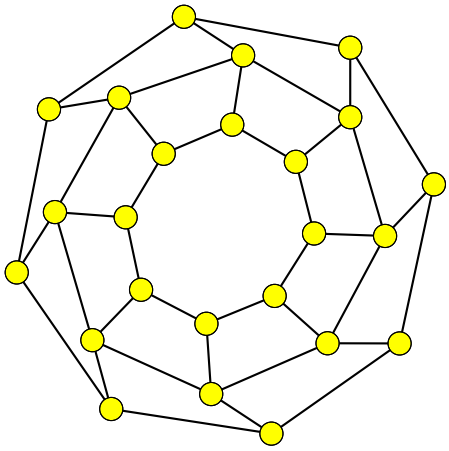
interaction	**potential**
$u, v \quad : \quad u, v \in V$	Repulsion (x_u, x_v)
$u, v \quad : \quad \{u, v\} \in E$	Attraction (x_u, x_v)

Introducing additional attraction potentials dragging the two cycles of degree three vertices, $V_{\mathsf{blue}}, V_{\mathsf{red}} \subseteq V$, to specified points, yields the following model.



interaction			potential
u, v	:	$u, v \in V$	Repulsion (x_u, x_v)
u, v	:	$\{u, v\} \in E$	Attraction (x_u, x_v)
v	:	$v \in V_{\text{blue}}$	Attraction $(x_v, \text{upper left})$
v	:	$v \in V_{\text{red}}$	Attraction $(x_v, \text{lower right})$

In a last example, the graph is flattened by adding a potential that penalizes crossings to the elementary combination of attraction and repulsion.



interaction			potential
u, v	:	$u, v \in V$	Repulsion (x_u, x_v)
u, v	:	$\{u, v\} \in E$	Attraction (x_u, x_v)
u_1, u_2, v_1, v_2	:	$\{u_1, v_1\} \neq \{u_2, v_2\} \in E$	Crossing $(u_1, v_1; u_2, v_2)$

We make extensive use of this flexibility in three quite different applications of graph visualization presented in the next sections.

3.2 Case Study I: Social Networks

In cooperation with members of the Faculty of Public Policy and Management at the University of Konstanz, we seek layout models for graphs arising from a social science methodology known as *social network analysis*. Since, at this stage, it is not known what effective visualizations of social networks look like, nor even how to assess the quality of a given visualization, we make extensive use of the inherent prototyping facilities of energy based placement.

Social network analysis is concerned with relationships among social entities, called *actors*. Relations may be affective, interactional, economic, political, organizational, or of any other type. Sometimes, "network" is just meant metaphorically, but in formal network analysis, networks are indeed considered as graphs. Structural variables measuring properties of the graph are used to support reasoning about the influence that structure is assumed to have on decision making or other consequences. Properties of the underlying structure are assumed to be determinants of outcomes. Foundations of the network perspective are given in Berkowitz (1982) and Burt (1982), and network analysis is discussed in Scott (1991). For an in-depth treatment see Wasserman and Faust (1994).

Our visualization efforts are guided by the principle that graphical design should be grounded on the substance to be communicated (see Section 2.1.2). Network visualization should be contingent on the general aim of social network analysis and its specific use of the "network analytic tool box"[5] for this purpose. We therefore first describe typical substance examined in social network analysis. Following some background on existing methods of social network visualization, we focus on one particular structural variable, centrality, and present a design depicting this substance while showing the network in its entirety.

3.2.1 Substance

The basic goal in the study of social networks is to describe the structure of relationships among a set of actors. A societal study using the network approach first delineates the set of relevant actors engaged, and then identifies the various relations of particular significance among them. The guiding idea behind this analytical perspective is that social configurations or developments can be explained by structured interaction within an actor set. Structuring is understood as an emergent effect which is restricting as well as enabling the actors to certain actions. Within the spectrum of network

[5]Kenis and Schneider (1991) in the context of policy networks, *i.e.* networks used to explain policy outcomes.

analytic methods, one can distinguish two types of structural perspectives, which are pursued at three different levels of detail: the actor, group, and network level.

On the one hand, a structural perspective may aim at detailed description of whether and how the different actors in a network are connected to each other via direct or indirect links. This viewpoint is called the connectedness perspective. On the other hand, the focus may be on similarity or dissimilarity of relational profiles of actors, which is then called the profile perspective. Actors with identical or nearly identical profiles are said to have equivalent network positions, and the distribution of roles and positions among actors is investigated. Often, both types of structural analysis are used in combination to analyze different aspects of an overall network.

At the actor level of analysis, structural variables measuring the relative position of actors based on their direct and indirect links to all other actors in the network are defined. Graph theoretic concepts like connectivity, path distance, or degree, are used to derive actor attributes assessing how central or how peripheral an actor is located, or what status an actor has in a link structure. Actor profiles are often defined on the basis the subgraph induced by their neighborhood.

At the group level, the analysis focuses on the question how a given network is structurally partitioned into subnetworks or groups. Partitioning can be performed from a connectedness perspective as well as from a profile perspective. Groups based on connectedness typically correspond to dense subgraphs. A number of related questions arise, asking for the number of groups, diameters, separators, or cut sizes. Operational concepts for cohesive groups include components, cliques, clans, and cores. Also of interest are special structural properties such as the existence of bridges or cutvertices ("brokers"). In contrast, groups based on profile similarity are identified via equivalence classes of actors, *i.e.* actors with identical or highly similar relations. Such a class is then interpreted as a social position representing a specific role with respect to the network. Social network analysis provides a spectrum of aggregation and division methods, such as block modeling or clustering procedures, to determine equivalence classes.

At the network level, structural analysis is concerned with different overall characteristics of the complete network structure such as how dense or how centralized a network is. Aggregate measures on the total network are particularly useful for comparison with other networks.

Since specific structural variables are used to analyze a given social network, they should be incorporated into a graphical design to produce meaningful visualizations. It is convenient to classify typically investigated substance into categories that help to better understand the possibilities and

limitations of a graphical design that is to convey it (see Table 3.1). These distinctions may serve as a guideline how visualizations can enhance the understanding of complex multidimensional settings by mapping different kinds of information to graphical variables on compatible levels of measurement.

The general distinction between semantic and syntactical substance (see Section 2.1.2) immediately applies to social networks. Essentially, semantic attributes are those closely related to the specific network under scrutiny (therefore, only examples of such attributes are given), while syntactic attributes are derived solely from the adjacency structure. The first three subcategories in both columns of Table 3.1 correspond to the three levels of aggregation: actor, group, and network. The fourth subcategory is introduced to account for properties that apply only in special cases.

In social network analysis, one is often interested in all three levels of aggregation simultaneously in order to explore or communicate information embedded in its context. Most desirable visualization techniques would therefore combine the associated perspectives in an information dense design that allows to switch between detail levels in a single image. In Tufte (1990), this is called micro/macro reading.

Syntactical Attributes	**Semantic Attributes**
Actor Attributes	**Actor Attributes**, *e.g.*
centrality	size of an organization
prestige / prominence	age of a person
Structural Partitions	**Attribute Partitions**, *e.g.*
cohesive subgroups	organizational subunits
structurally equivalent actors	legal form of an organization
role equivalent actors	attitudes towards policy issues
Network Structure	**Network Attributes**, *e.g.*
size	period of data gathering
density	reliability
centralization	
cohesiveness	
Structural Positions	**Selected Attributes**, *e.g.*
bridge	distinct institutional role
broker	

Table 3.1: Typical substance analyzed in social network analysis

3.2.2 Social Network Visualization

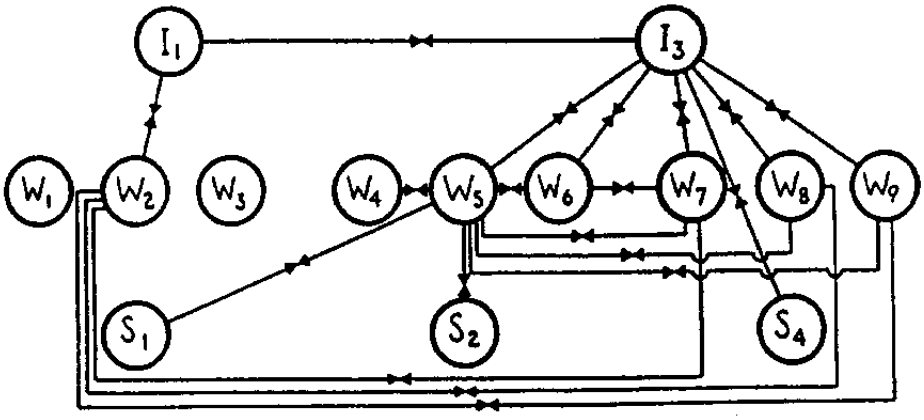
Given the fact that graphical presentations were among the very first means to represent and analyze social structures (Moreno, 1953), it is astonishing how little attention the subject was paid subsequently. We give a brief summary of the history of network visualization and of methods currently in use. More detail on the early stages of network visualization is provided in Klovdahl (1981), one of the rare publications on social network visualization so far. Although interesting and very to the point, it seems to have largely been ignored.[6]

Already in the 1930s, Moreno used graphical presentations of relational sociological data. His *sociograms* (point-and-line representations) comprise one of the earliest formalizations of social structure, influencing – directly or indirectly – a number of subdisciplines of the social sciences like social psychology, social anthropology, sociology of organization, *etc.*According to Scott (1991, p. 18), one of the major investigations popularizing this approach is the one known as the "Hawthorne Study" (Roethlisberger and Dickson, 1939). In this study, social relations among workers in a bank wiring room are described by sociograms. Actors are represented by circles placed on horizontal lines indicating the actor's occupation, and the presence of a relationship of certain type is shown by an arrow. Figure 3.2(a) is an example of such a diagram. It is an interesting observation that these diagrams resemble circuit schematics and wiring plans. One has an immediate idea who was asked to draw them.

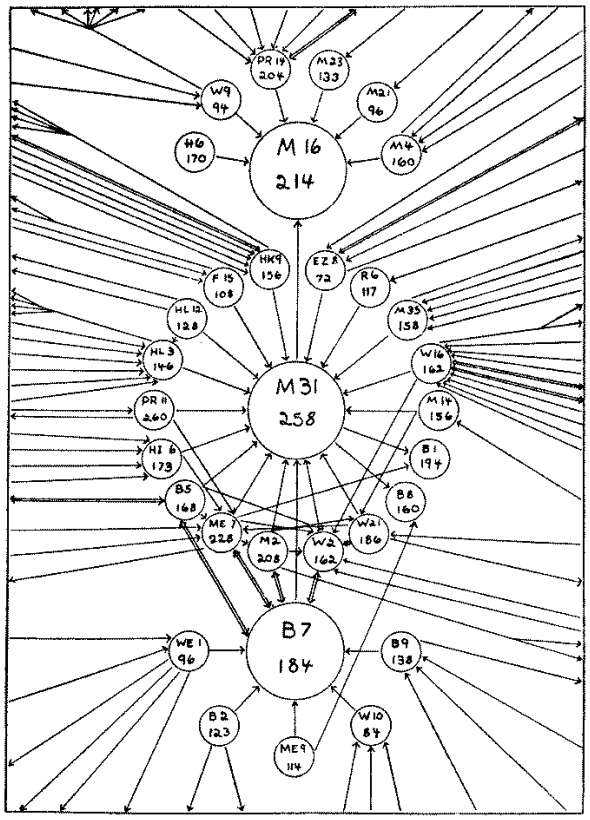
Although these early forms of graphical presentation were applied in structural analyses at all levels of society, and although they were considered a fruitful method of exploration, it seems that the tedious process of manually designing network visualizations has prohibited major innovations until the use of computers became customary. Nevertheless, the few examples of visual representations also include other specialized variants of the general sociogram. One such example, analyzed in Klovdahl (1981) and shown in Figure 3.2(b), is a diagram from Lundberg and Steele (1938). In this straight-line representation a focal actor is placed in the center of the layout area. To reduce visual clutter, the other actors are placed manually such that the number of crossings remains small.

Instead of an *a-priori* important actor, structural variables govern the design of the *target diagram* from Northway (1940, see Figure 3.3). Actors are clustered according to quartiles of acceptability scores they received in questionnaires. The vertices of each cluster are mapped to one of four concentric

[6] According to the Social Science Citation Index, it was cited only four times from 1981 to 1996.



(a) Roethlisberger and Dickson (1939, Figure 44)



(b) Lundberg and Steele (1938, Chart I)

Figure 3.2: Historical examples of sociograms

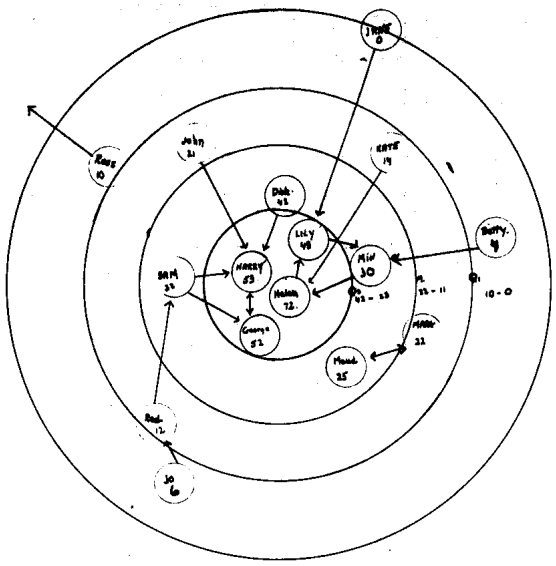


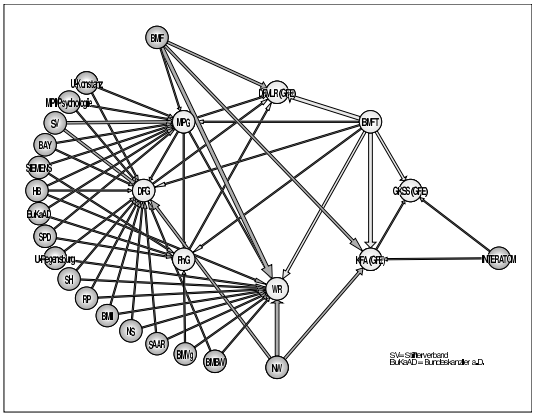
Figure 3.3: Target diagram (Northway, 1940, p. 149)

rings, where the lowest quartile is farthest from the center.[7] This approach is extended in Section 3.2.3. Finally, a network representation used quite often is the *circle diagram*, in which a sociogram is constructed around the circumference of a single circle with edges drawn as chords. This kind of layout is attributed to "make the pattern of lines more visible" (Scott, 1991, p. 149), but Blythe *et al.* (1996) provide empirical evidence that it is rather confusing.

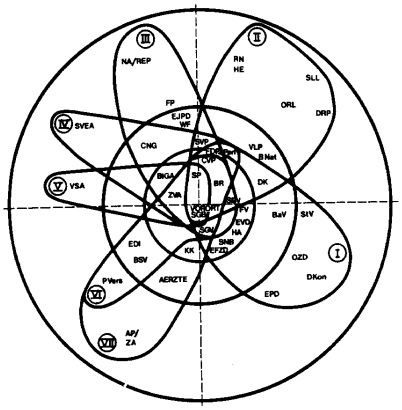
Increasing availability of software tools, and the use of statistical methods like multidimensional scaling (MDS) or cluster analysis initiated new concepts and techniques of visualization. Therefore, sociograms today are accompanied by three other forms of graphical presentation that do in turn focus on particular aspects of the input network: MDS scatterplots representing path distances or structural similarity in terms of Euclidean distances, dendrograms depicting clustering hierarchies, and Venn diagrams showing cluster overlap. Figure 3.4 gives one example for each design.

Unfortunately, current visualizations of networks in sociograms are limited in the sense that they try to make use of what is available, rather than to state what is desired and ask for tools implementing these requirements. Recent work on visualization (Freeman, 1996, 1997) orients itself towards the applicability and usefulness of existing computer software. Consequently,

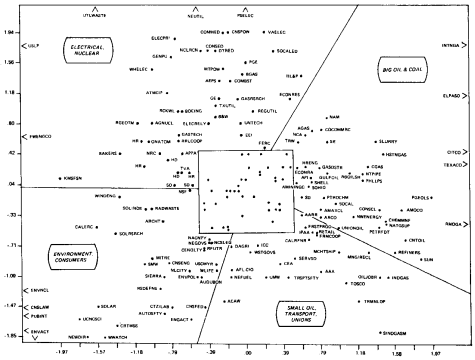
[7]Northway (1940, p. 149) reports that the actual placement within the rings was determined by arranging poker chips on a table.



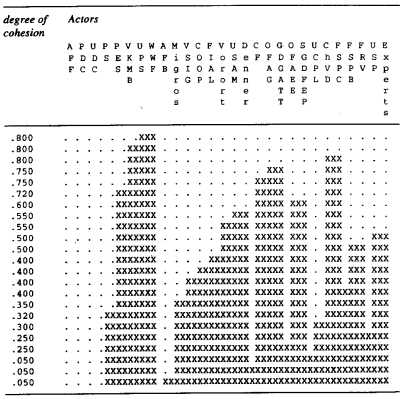
(a) Sociogram (Mayntz, 1994, p. 38)



(b) Venn diagram (Kriesi, 1982, p. 147)



(c) MDS scatterplot (Laumann and Knoke, 1987, p. 242)



(d) Dendrogram (Scarini, 1996, p. 98)

Figure 3.4: Graphical presentations commonly used for network data

current methods are almost oblivious of what is described in Section 2.1.2 to be the three building blocks of information visualization (substance, design, algorithm).

Currently used network visualization tools fall into four categories: General purpose graphical editors, two kinds of drawing programs developed for other than social networks, and tools and strategies explicitly meant to be used for social networks.

General purpose graphical editors. They are the least comfortable, yet most flexible tools, and are available for virtually every computer platform. The latter is an important criterion for production techniques to become sufficiently widespread. Such editors do provide a rich set of editing functions, but almost no features tailored to networks. In particular, vertices representing actors and edges representing links can be moved independently on the screen, because such programs obviously do not know about structural issues. The burden of specifying and implementing a graphical design is put on the user. Moreover, manual design of meaningful graphical presentations even for networks of moderate size is a tedious, if not intractable, task. The latter explains why tabular presentation is often preferred.

Network drawing software from other disciplines. There are two such categories. One consists of those programs that draw specific networks other than social networks, like, for instance, molecule structures.[8] These provide means to layout a network in domain dependent ways. For example, atoms and bonds of a molecule are positioned according to underlying energy laws. Obviously, such programs do not account for the substance of a social network. And because of that, it is only by chance that they specify an effective design. Their usefulness seems to be limited to the fact that they, in general, produce drawings that are pleasing from an ergonomic point of view.

The other such category consists of general purpose graph drawing software. It mainly contains domain independent graph layout tools offering a variety of algorithms for different layout models. Examples are, of course, fairly general graph drawing systems like GRAPHLET (Himsolt, 1996) or DAVINCI (Fröhlich and Werner, 1995). Since the design principles implemented by these algorithms are usually ergonomic requirements like small area, small number of edge crossings, or small number of bends on the edges that apply

[8] MOVIEMOL (`http://chem-www.mps.ohio-state.edu/~lars/moviemol.html`) is a popular example.

to almost any kind of graph visualization, they also do not account for the specific substance of social networks.

Software and strategies for social networks. This category comprises plots produced by routines in analytical software packages for multidimensional scaling (see, *e.g.*, Kruskal and Wish, 1978)) or eigen analysis (see, *e.g.*, Mohar, 1991; Richards and Seary, 1997), as well as designated drawing programs for social networks, and the only living variation of the general sociogram that could be tracked down, circle diagrams.

Multidimensional scaling and eigen analysis are arguably the network visualization strategies for which substance, design, and algorithm are most clearly identified. Both produce scatterplots: MDS plots reflect proximity in higher dimensional data (*e.g.* path distances) in fewer dimensions, while eigen analysis plots are produced according to eigenvectors of matrices representing the network, *e.g.* the adjacency matrix or the Laplacian matrix (see Section 4.1.1). Both methods display particular aspects of a network's overall substance, but typically lead to unpleasing drawings when edges are shown in the scatterplot. The *stress value* (Kruskal and Wish, 1978) of an MDS provides a measure of how well the plot fits the design (mapping proximity to Euclidean distances).

Drawing programs like KrackPlot (Krackhardt *et al.*, 1994), Pajek (Batagelj and Mrvar, 1998), or MultiNet[9] are the most advanced tools available today. Besides actor positioning according to MDS and eigendecomposition, respectively, the first two also include refinements of the spring embedder. While the layout of KrackPlot is based on the model of Davidson and Harel (1996), Pajek includes the models of Fruchterman and Reingold (1991) Kamada and Kawai (1989). In both cases the design is a function of the algorithm rather than the substance. However, it is interesting to note that applying these algorithms nevertheless appears to be very reasonable if the substance of interest is proximity in terms of path distances. The implied design is closely related to the design of MDS with some additional ergonomic criteria, *e.g.* nodes being distributed more evenly in the layout space. A similar layout algorithm is used by Krempel (1997).

The idea of circle diagrams clearly defies any definition of substance. It is a design purely based on a doubtable ergonomic criterion, simplification through prescribed shape (cf. Blythe *et al.* 1996; McGrath *et al.* 1997). An extension presented in Krempel (1993) requires actors to be placed such that the total length of connecting lines is small. Even though dense subgraphs then tend to cluster in small arcs, there is no precise definition of the sub-

[9] `http://www.sfu.ca/~richards/multinet.htm`

stance thus revealed. Not to speak of evidence that this design is effective.

In summary, existing methods for social network visualization do not clearly identify substance, design, and algorithm at the same time. The only exceptions to this observation are diagrams resulting from designated analytical tools like multidimensional scaling or eigen analysis. However, they disregard ergonomic aspects and have a very limited definition of a network's substance. Often they only show the aggregated values, but not the network itself. Because of the relation between the objective function of Kamada and Kawai (1989) and MDS pointed out earlier, energy based placement may be capable of integrating ergonomic and other criteria in an effective design. It might turn out that criteria often present in energy based placement approaches (in particular uniform node distribution and uniform edge lengths) work well enough in displaying many relevant aspects of a network, like symmetry, cohesive subgroups,[10] brokers, and so on. For now, this remains an open question that can only be answered by careful analysis of the substance to be displayed and the substance perceived. The following section provides a first step in this direction.

3.2.3 Depicting Centrality

Our general assumption is that, in order to routinely produce effective visualizations of social networks, standards for effective design on the basis of a network's substance are needed. Layout is the most challenging part of such guidelines, since many reasonable layout criteria are only approximately satisfiable,[11] or conflicting. A possible approach to resolve the latter difficulty at least partially is to "triangulate" different analytical perspectives.[12] Each of them should provide greater accuracy in the description of selected aspects, while their combination diminishes the risk of being taken by methodological artifacts. We here take on one such perspective and use energy based placement to prototype layout models accordingly.

[10] See McGrath *et al.* (1998) on the perception of groups.

[11] For example, one of Tufte's preconditions for graphical excellence is to tell the truth about the data (Tufte, 1983, p. 51). A layout requirement might therefore be that all edges representing the same type of relation be of equal length. However, such layouts do not exist for all graphs, and it is $\mathcal{NP}$-hard to decide, whether it exists or not (Johnson, 1982).

[12] *Triangulation* is generally defined as the combination of methodologies in the study of the same phenomenon. It stems from a military metaphor pointing to the use of multiple reference points to locate an object's exact position (Jick, 1979, p. 602). For an implicit use of triangulation in network analysis see Doreian (1988).

Part of an overall substance, the notion of centrality is one of the most important aspects in social network analysis. It is investigated on all three levels of detail, and therefore distinguished into *actor centrality*, *group centrality*, and *network centralization*. It appears to be the ideal starting point to experiment with layout models highlighting structural variables, since being central has an intuitive geometric connotation.

Several non-equivalent operationalizations of the centrality concept on the actor level are considered in network analysis. They are mostly based on vertex degrees or derived from pairwise distances. Freeman (1979) outlines a framework to obtain a normalized and a network level measure from a given actor level centrality index. If $C(v)$ is a non-negative actor level centrality index, then the normalized centrality measure

$$C'(v) = \frac{C(v)}{\max\limits_{G_n} \max\limits_{w \in V_n} C(w)}$$

where maximization is over all graphs $G_n = (V_n, E_n)$ on $n = |V|$ vertices, takes values between zero and one. We will see below that, for some centrality measures, it is advantageous to use this normalized version in layout design. The extreme case of a centralized network is assumed to be such that there is a highly central actor, while all others are peripheral. A network centralization index can therefore be defined by

$$C(G) = \frac{\sum_{v \in V} \left(\max_{u \in V} C'(u) - C'(v)\right)}{\max\limits_{G_n} \sum_{v \in V_n} \left(\max_{u \in V_n} C'(u) - C'(v)\right)}$$

where maximization is again over all connected undirected graphs $G_n = (V_n, E_n)$ with the same number n of vertices as G. In the following, we apply this framework to centrality indices based on four different measures: degree, closeness (Sabidussi, 1966), eccentricity (Hage and Harary, 1995), and betweenness (Freeman, 1977). Note that eccentricity is not treated in Freeman (1979). Let $G = (V, E)$ be an undirected and connected graph with n vertices and m edges throughout.

Degree. A straightforward notion of centrality reflecting the activity of an actor is the number of links he or she maintains. *Degree centrality* is hence defined to be the degree of the corresponding vertex,

$$C_D(v) = d_G(v)$$

Using the maximum degree possible in any graph on n vertices, normalization yields

$$C'_D(v) = \frac{C_D(v)}{n-1}$$

Since the largest sum of differences between a vertex of maximum degree and all others in a connected graph occurs if the latter have degree one, the *degree centralization* of a network equals

$$\begin{aligned} C_D(G) &= \frac{\sum_{v \in V} \left(\frac{\Delta(G)}{n-1} - C'_D(v) \right)}{\max_{G_n} \sum_{v \in V_n} \left(\frac{\Delta(G_n)}{n-1} - C'_D(v) \right)} \\ &= \frac{\sum_{v \in V} \left(\Delta(G) - d_G(v) \right)}{\max_{G_n} \sum_{v \in V_n} \left(\Delta(G) - d_G(v) \right)} \\ &= \frac{\sum_{v \in V} \left(\Delta(G) - d_G(v) \right)}{(n-1) \cdot (n-1-1)} \\ &= \frac{\sum_{v \in V} \left(\Delta(G) - d_G(v) \right)}{n^2 - 3n + 2} \end{aligned}$$

where $\Delta(G)$ is the maximum degree of any vertex in G.

Closeness. The degree of a vertex is a very local concept of centrality, counting how many others are immediately linked to an actor. The underlying notion of accessibility can be expanded to the whole graph using the sum of distances, or *closeness*,

$$c_G(v) = \sum_{w \in V} d_G(v, w)$$

of a vertex to all others. The closer to all others, the more central an actor is. *Closeness centrality* is hence defined as inverse sum of distances,

$$C_C(v) = \frac{1}{c_G(v)}$$

The maximum possible value occurs for vertices directly connected to all others. It follows that the relative closeness centrality,

$$C'_C(v) = \frac{C_C(v)}{1/(n-1)} = \frac{n-1}{\sum_{w \in V} d_G(v, w)}$$

is the inverse average distance of a vertex from all others. The maximum sum of differences in relative centrality scores is attained in a star, so that

closeness centralization evaluates to

$$\begin{aligned} C_C(G) &= \frac{\sum_{v\in V}(\max_{u\in V} C'_C(u) - C'_C(v))}{(n-1)\cdot\left(\frac{n-1}{n-1} - \frac{n-1}{1+(n-2)\cdot 2}\right)} \\ &= \frac{\sum_{v\in V}(\max_{u\in V} C'_C(u) - C'_C(v))}{(n^2-3n+2)/(2n-3)} \\ &= \frac{\sum_{v\in V}(\max_{u\in V} C_C(u) - C_C(v))}{(n-2)/(2n-3)} \end{aligned}$$

Eccentricity. In general, closeness centrality counts connections used to reach other actors more than once. If the interest is solely in the longest distance, centrality can be based on the *eccentricity*

$$e_G(v) = \max_{w\in V} d_G(v,w)$$

of a vertex. An actor is considered central, if the eccentricity of its corresponding vertex is small. Note that the vertices with minimum eccentricity form the graph theoretic center of the graph. We thus define *graph centrality* by

$$C_G(v) = \frac{1}{e_G(v)}$$

and its maximum value is one. It follows that $C'_G(v) = C_G(v)$, and

$$\begin{aligned} C_G(G) &= \frac{\sum_{v\in V}(\max_{u\in V} C'_G(u) - C'_G(v))}{(n-1)\cdot(1-\frac{1}{2})} \\ &= \frac{\sum_{v\in V}(\max_{u\in V} C'_G(u) - C'_G(v))}{(n-1)/2} \end{aligned}$$

Betweenness. For any three distinct vertices u, v, w let $P(u,w)$ be the set of shortest paths between u and w, and $P_v(u,w)$ be the set of shortest paths between u and w passing through v. Then, the sum of ratios

$$b_G(v) = \sum_{u\neq v\neq w} \frac{|P(u,w)|}{|P_v(u,w)|}$$

of shortest paths that v sits on is called *betweenness* of v. Betweenness reflects the extent to which an actor is a mediator in the network. As a measure of *betweenness centrality*, we simply use $C_B(v) = b_G(v)$. Its maximum value is attained by the central vertex of a star, so that

$$C'_B(v) = \frac{C_B(v)}{(n-1)(n-2)/2}$$

Stars also yield the maximum sum of differences in relative betweenness scores, since the central vertex has relative betweenness centrality one, and all other vertices score zero. It follows that

$$\begin{aligned} C_B(G) &= \frac{\sum_{v\in V}\left(\max_{u\in V} C'_B(u) - C'_B(v)\right)}{n-1} \\ &= \frac{\sum_{v\in V}\left(\max_{u\in V} C_B(u) - C_B(v)\right)}{n^3-4n^2+3} \end{aligned}$$

The above four families of measures in general differ both in actor centrality and network centralization scores. However, they all assign minimum centralization to complete graphs, and maximum centralization to stars. Freeman (1979) discusses the advantages and disadvantages of using degree, closeness, or betweenness centrality in different research settings.

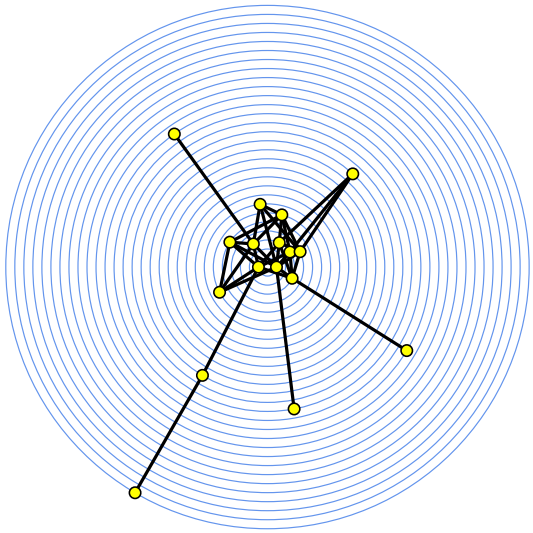
We argued that graphical design should be based on the substance to be communicated. To depict centrality in straight-line representations of social networks, we intend to place vertices at some distance from the center of the diagram that reflects their centrality score, thus drawing on the geometric intuition behind centrality.

Let $r(v)$, $v \in V$, be specified radii. Then, we constrain the set of feasible positions of vertex v to those at distance $r(v)$ from the origin, which is presumed to be the center of the diagram. Adding some potentials for ergonomics, we obtain a generic *centrality layout model*

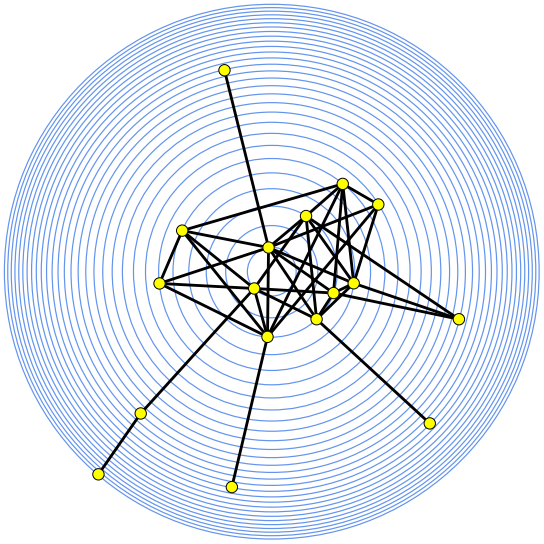
interaction			**potential**
u, v	:	$\{u, v\} \in E$	Distance $(x_u, x_v \mid c_\delta(u, v))$
u, v	:	$\{u, v\} \notin E$	Repulsion (x_u, x_v)
u, v, w	:	$v \notin \{u, w\} \in E$	Repulsion $(x_v; x_u, x_w)$
u_1, u_2, v_1, v_2	:	$\{u_1, v_1\} \neq \{u_2, v_2\} \in E$	Crossing $(u_1, v_1; u_2, v_2)$
dependency			**constraint**
v	:	$v \in V$	$d(x_v, \langle 0, 0\rangle) = r(v)$

which allows to specify a mapping of vertices to concentric circles by choosing radii $r(v)$. Note that the distance parameter $c_\delta(u, v)$ is made dependent on the pair of adjacent vertices. In general, edge lengths are desired to roughly equal the difference in radius, $|r(u)-r(v)|$. If this difference is small, however, $c_\delta(u, v)$ is increased to prevent very short edges.

Figure 3.5(a) shows a network of local organizations in a German city that are involved in drug policy making (Kenis, 1999). The network defining relation is private communication between these organizations. A structural



(a) closeness



(b) normalized closeness centrality

Figure 3.5: Closeness levels cause a resolution problem which is resolved by using normalized closeness centrality

variable of interest is closeness centrality, which is depicted by choosing radii according to the closeness of each vertex. For ease of notation, let $|\min f|$ and $|\max f|$ denote the number of minima and maxima of a function f, respectively. The most central actor should be placed in the center of the diagram, but since there may be more than one such actor, radii are defined by

$$r(v) = c_G(v) - \min_{u \in V} c_G(u) + |\min_{u \in V} c_G(u)| - 1$$

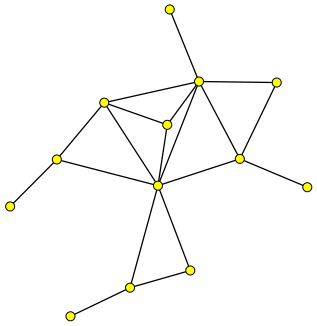
The fact that closeness yields large differences in value especially for less central actors causes some obvious drawbacks. While edges to peripheral actors become quite long, the set of central actors clusters in a small area around the center. We can reverse this effect using closeness centrality instead of closeness. Normalized closeness centrality is bounded from above by one, so that taking $1 - C'_C(v)$ ensures that the relative ordering of distances matches our intuition. To avoid overlaps caused by more than one vertex having radius zero, and to scale the diagram to fit into a standardized box, we actually use

$$r(v) = 1 - \frac{C'_C(v) - \min_{u \in V} C'_C(u)}{\max_{u \in V} C'_C(u) - \min_{u \in V} C'_C(u) + c(|\max_{u \in V} C'_C(u)|)}$$

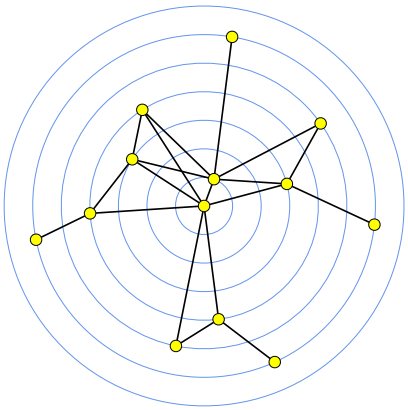
where $c(|\max_{u \in V} C'_C(u)|)$ is an offset depending on the number of maximally central actors. See Figure 3.5(b) and note the resulting non-linear scaling of closeness levels.

The idea generalizes to other centrality measures using straightforward scalings. Figure 3.6 shows centrality layouts based on the four measures defined above, and a layout obtained from an implementation of a typical spring embedder variant (Fruchterman and Reingold, 1991) implemented in LEDA (Mehlhorn and Näher, 1999). All subfigures show the same graph, which is the communication network of local drug policy organizations in another German town. Levels supporting comparison of centrality scores within a diagram are drawn for each possible integer value of degree, eccentricity, and closeness, respectively. In centrality layouts based on betweenness, only occurring levels are shown.

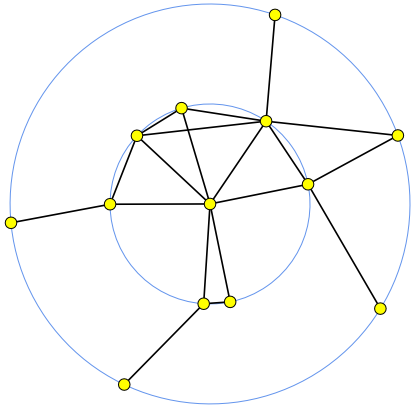
Social network data is often gathered from questionnaires. A frequently encountered problem is that links claimed by one actor are not reciprocated by the other. The design used to produce Figure 3.7 includes directed edges to additionally show who claimed to communicate with whom without being confirmed. Closeness centrality scores are computed based on the reciprocated links, which form the subnetwork already presented in Figure 3.6. Actors without any reciprocated link are placed outside of all closeness levels shown. More substance is coded in graphical variables color (attitude



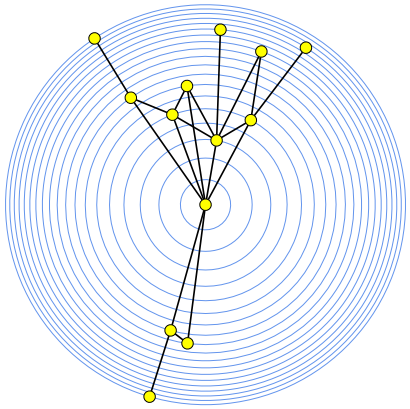
(a) no centrality constraints



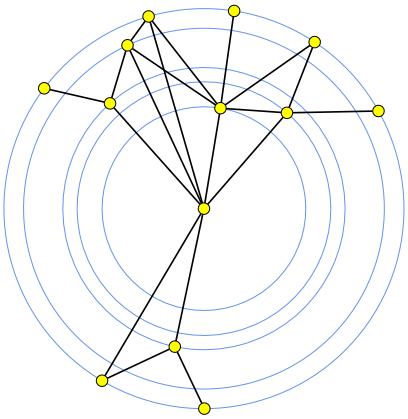
(b) degree centrality



(c) graph centrality



(d) closeness centrality



(e) betweenness centrality

Figure 3.6: Layouts based on various centrality measures

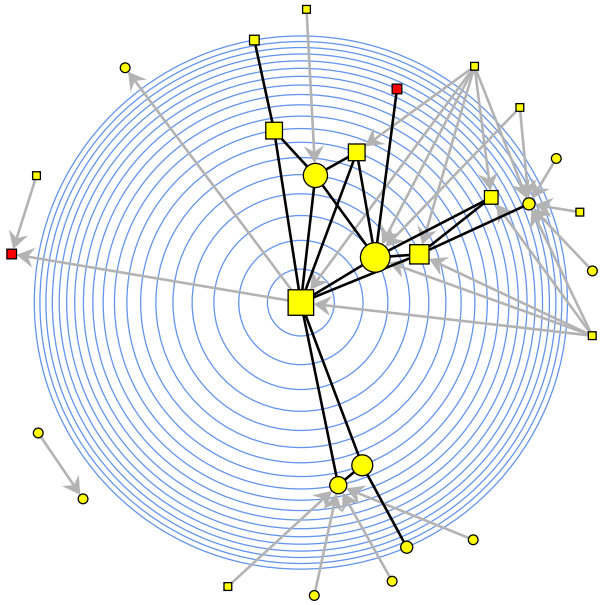


Figure 3.7: Providing additional insight by showing unreciprocated links. Centrality levels are based on reciprocated links only, but all edges contribute to the layout energy function. Size indicates degree centrality, while color and shape code semantic attributes of vertices

towards drug users), shape (governmental or private organization), and size (degree centrality). Note that two (governmental) actors are peripheral, because none of their claimed links are reciprocated, while two (privately run) organizations are fairly peripheral, because they confirm only few links. It may be interesting to note that the underlying undirected graph was already shown in Figure 2.5.

Similarly, the network of actors involved in a privatization process in Eastern Germany (Raab, 1999) shown in Figure 3.8 has a number of unreciprocated links. The relation represents strategic cooperation. It is difficult to judge actor centrality from a spring embedding, which is commonly used in social network visualization today. For example, it is unclear whether the vertices represented by smaller boxes close to the center of the diagram are central in the network. The question is immediately resolved by looking at the centrality layout. Because of case specific considerations, radii for the centrality layout model are determined from closeness centrality scores in the underlying undirected network. However, the ratio of height to width corresponds to the ratio of in- to outdegree, providing graphical evidence that the most central actors are central because they say so themselves.

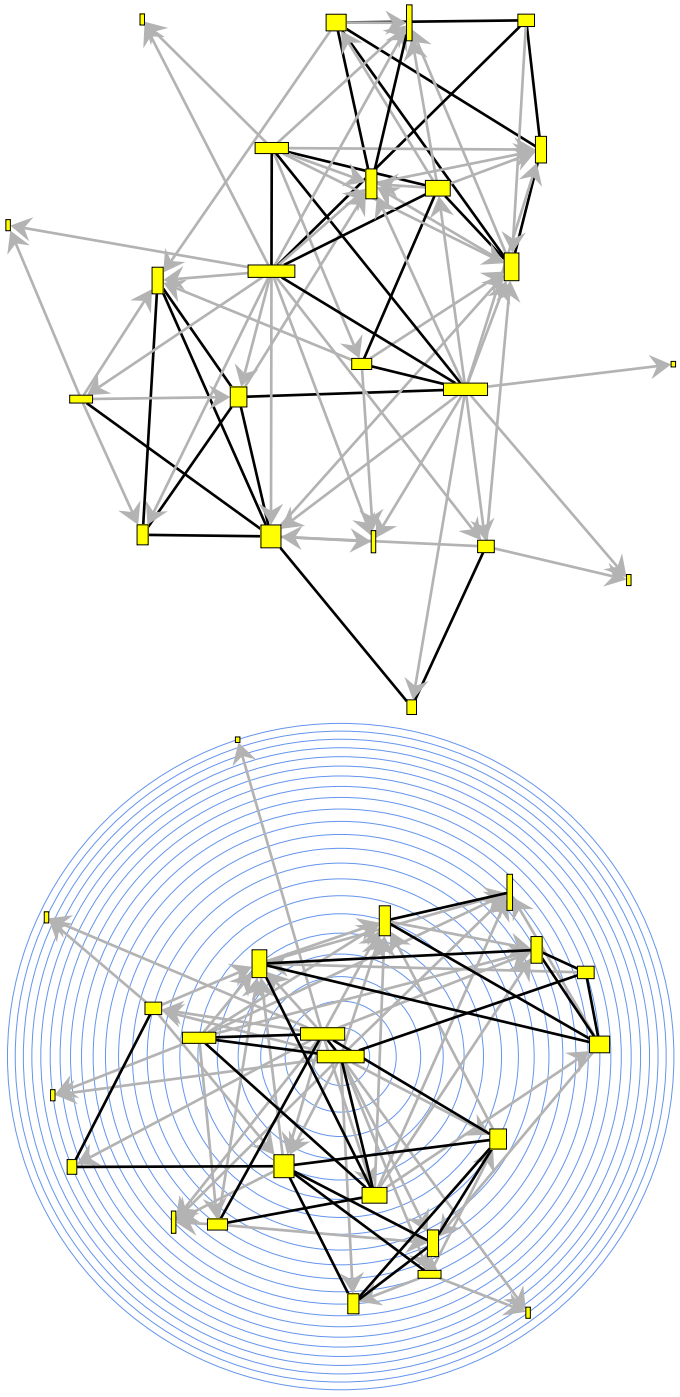


Figure 3.8: Spring embeddings do not properly communicate centrality

3.3 Case Study II: Dynamic Web Structures

Graph A of the 1998 Graph Drawing Contest (Eades *et al.*, 1998) is a welcome opportunity to study dynamic layout models for straight-line representations. Since 1993, a contest is organized along with the annual *Symposium on Graph Drawing* (Di Battista *et al.*, 1993a; Tamassia and Tollis, 1995; Brandenburg, 1996; North, 1996b; Di Battista, 1997; Whitesides, 1998). It usually features graphs from different applications posing distinct problems with respect to graphical presentation. Reports on earlier contest are given in Eades and Marks (1995, 1996) and Eades *et al.* (1996, 1997). In the present case, a dynamic graph of links between World Wide Web pages is given as a list of link additions and deletions. Figure 3.9 shows the textual data presentation provided for the contest. The substance of the graph is implied by the fact that contestants should

> " ... depict the content and structure of the graph as it evolves."

As in the case of social networks, we here do not consider the content of the graph (semantic substance) for layout, but rather focus on its evolving structure (syntactic substance).

Because the third dimension provides an additional degree of freedom, we choose a graphical design that animates a straight-line representation of the graph in space. In particular, the near-hierarchical structure typical for the pages of a single web site is mapped to the above/below relation. For graceful animation, layouts before and after each modification must depend on each other. The contest graph is hence a neat opportunity to study the integration of stability terms in layout objective functions advocated in Section 2.2.2.

A drawback of using the third dimension is that we have to carefully select a point from which to view the graph, because otherwise parts of the structure may not be recognizable. We outline a simple, yet satisfactory heuristic in Section 3.3.2. Some remarks on how the animation is actually carried out and a number of sample frames conclude this section.

3.3.1 Dynamic Hierarchical Layout

The layout model is described in three steps. First, static structure is mapped into space, then stability is introduced to limit changes between consecutive layouts, and finally layouts are sequenced with respect to graph modifications.

Static layout model. The static layout model should map structural information contained in the set of links to spatial relations in three dimensions.

```
add www.att.com/catalog/consumer -> www.att.com
add www.att.com/att -> www.att.com/catalog/consumer
add www.att.com -> www.att.com/catalog/consumer
add www.att.com/catalog/consumer -> www.att.com/cmd/custcare
add www.att.com/catalog/consumer -> www.att.com/write
add www.att.com -> www.att.com/worldnet
add www.att.com/catalog/consumer -> www.att.com/terms.html
add www.att.com/att -> www.att.com/learningnetwork
add www.att.com/catalog/consumer -> www.att.com/cgi-bin/ppps.cgi
add www.att.com -> www.att.com/catalog/small_business
add www.att.com/whatsnew -> www.att.com/catalog/consumer
add www.att.com/catalog/small_business -> www.att.com/cgi-bin/bmd_cart.cgi
add www.att.com/catalog/consumer -> www.att.com/cmd/jump
add www.att.com/att -> www.att.com/catalog
add www.att.com/textindex.html -> www.att.com/catalog/small_business
add www.att.com/catalog/small_business -> www.att.com/bmd/tollfree
add www.att.com/whatsnew -> www.att.com/worldnet/wmis
add www.att.com/catalog/consumer -> search.att.com
add www.att.com/catalog/small_business -> www.att.com/bmd/jump
add www.att.com/features -> www.att.com/rock
add www.att.com/catalog/small_business -> www.att.com
add www.att.com/home -> www.att.com/catalog/consumer
delete www.att.com -> www.att.com/catalog/consumer
add www.att.com/catalog/small_business -> www.att.com/services
delete www.att.com/att -> www.att.com/learningnetwork
delete www.att.com/catalog/consumer -> www.att.com/cmd/custcare
add www.att.com/catalog/small_business -> www.att.com/write
add www.att.com/net -> www.att.com/worldnet/wis/sky/signup.html
add www.att.com/catalog/small_business -> www.att.com/bmd/products
add www.att.com/catalog/small_business -> search.att.com
add www.att.com/net -> www.att.com/worldnet/wmis
add www.att.com/catalog/small_business -> www.att.com/terms.html
delete www.att.com/catalog/small_business -> www.att.com/bmd/tollfree
add www.att.com/att -> www.att.com/catalog/small_business
add www.att.com/news -> www.att.com/catalog/consumer
add www.att.com/whatsnew -> www.att.com/catalog/small_business
add www.att.com/net -> www.att.com/worldnet/intranet
delete www.att.com/whatsnew -> www.att.com/catalog/consumer
add www.att.com/catalog/small_business -> www.att.com/bmd/custcare
add www.att.com/catalog/consumer -> www.att.com/cgi-bin/cart.cgi
delete www.att.com/home -> www.att.com/catalog/consumer
add www.att.com/catalog/small_business -> www.att.com/cmd
add www.att.com/catalog/consumer -> www.att.com/services
add www.att.com/worldnet -> www.att.com/worldnet/intranet
add www.att.com/services -> www.att.com/catalog
delete www.att.com/catalog/consumer -> www.att.com
add www.att.com/services -> www.att.com/catalog/consumer
delete www.att.com/catalog/consumer -> www.att.com/services
add www.att.com/services -> www.att.com/catalog/small_business
add www.att.com/whatsnew -> www.att.com/news
delete www.att.com/whatsnew -> www.att.com/worldnet/wmis
add www.att.com/catalog/consumer -> www.att.com/cmd/products
delete www.att.com/net -> www.att.com/worldnet/wis/sky/signup.html
delete www.att.com/catalog/small_business -> www.att.com/services
add www.att.com/speeches -> www.att.com/speeches/index96.html
add www.att.com/worldnet -> www.att.com/worldnet/wmis
delete www.att.com/att -> www.att.com/catalog
add www.att.com/textindex.html -> www.att.com/catalog/consumer
delete www.att.com/services -> www.att.com/catalog
add www.att.com/news -> www.att.com/catalog/small_business
add www.att.com/international -> www.att.co.uk
delete www.att.com/catalog/small_business -> search.att.com
add www.att.com/business -> www.att.com/catalog/small_business
delete www.att.com/textindex.html -> www.att.com/catalog/consumer
add www.att.com/write -> www.catalog.att.com/cmd/jump
```

Figure 3.9: Data given for Graph A of the 1998 Graph Drawing Contest

Suppose we are given only one directed graph $G = (V, E)$. The layout elements in a three-dimensional straight-line representation are only the vertices of the graph, $L = V$, and we assume that every grid point in a reasonably large box centered at the origin is a feasible position.

For readability we stick to the proven local criteria described in Section 3.1. Namely, we use repulsion potentials between every pair of vertices and every pair of an edge and a non-incident vertex. Attraction potentials are used between adjacent vertices. Note that there is no need for additional potentials to avoid crossings.

The contest graph is disconnected at several points in time, so that the repulsion potentials would cause components to drift arbitrarily far apart. Additional attraction potentials therefore tie one vertex of each connected component of the graph to the origin of the layout space. This vertex is called the *representative* of its component.

Since the contest graph consists of web pages and changing links between them, it is a general directed graph, yet with a tendency to contain hierarchical acyclic substructures. This is precisely the syntactic substance of the graph. To encode the direction of edges graphically, we favor them pointing downward. In other words, the z-coordinate of the start vertex of any directed edge should be larger than of its end vertex. Sugiyama and Misue (1995) introduce rotative forces to align edges with an imaginary magnetic field (see Section 3.1). We adapt this idea using potentials

$$\mathsf{Rotation}\left(x_u, x_v \mid c_\delta^2\right) = c_\delta^2 \cdot \left(\frac{\arccos \dfrac{x_u^{(3)} - x_v^{(3)}}{d(x_u, x_v)}}{\dfrac{\pi}{2}} \right)^2$$

that measure the angle between a directed edge $(u, v) \in E$ and the z-axis (cf. Figure 3.10). Here, $x_v^{(3)}$ denotes the z-coordinate of a position $x_v = \langle x_v^{(1)}, x_v^{(2)}, x_v^{(3)} \rangle \in \mathcal{X}_v$, $v \in V$. Standardization with $\pi/2$ and multiplication with the root of a distance constant c_δ^4 ensure that edge length and downward direction have comparable influence on the objective function. The static layout model for 3D straight-line representation of $G = (V, E)$ is summarized as follows

interaction			potential
u, v	:	$u \neq v$	$\mathsf{Repulsion}\,(x_u, x_v \mid c_\delta^4)$
u, v	:	$\{(u, v), (v, u)\} \cap E \neq \emptyset$	$\mathsf{Attraction}\,(x_u, x_v)$
u, v	:	$(u, v) \in E$	$\mathsf{Rotation}\,(x_u, x_v \mid c_\delta^2)$
v	:	v representative of a component	$\mathsf{Attraction}\,(x_v, \langle 0, 0, 0 \rangle)$
u, v, w	:	$u \notin \{v, w\}, \{(v, w), (w, v)\} \cap E \neq \emptyset$	$\mathsf{Repulsion}\,(x_u; x_v, x_w \mid c_\delta^4)$

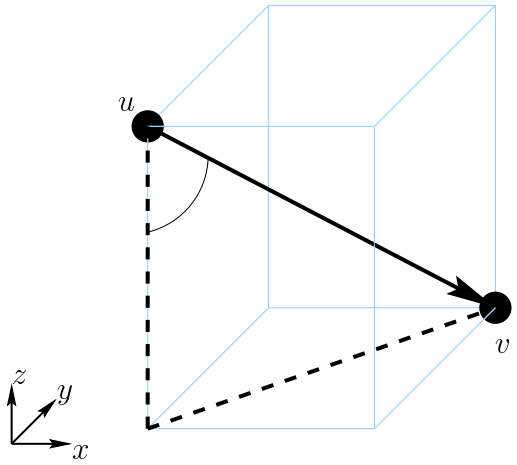


Figure 3.10: The angle between the directed edge $(u, v) \in E$ and the z-axis equals $\arccos \frac{x_u^{(3)} - x_v^{(3)}}{d(x_u, x_v)}$.

Note that c_δ controls the desired edge length, while a pair of arcs (u, v) and (v, u) yields two conflicting potentials causing u and v to have roughly the same z-coordinate.

Incorporating stability. In order to introduce dependencies between consecutive layouts, assume now that we are given a directed graph $G = (V, E)$ with a partial layout $y = (y_v)_{v \in \Lambda \subseteq V}$. The partial layout reflects the fact that vertices of the graph not introduced during the most recent modification are already positioned in space. Within the Bayesian framework of Section 2.2.2, the partial layout is considered an observation for which a deviation measure *i.e.* a criterion of stability, ought to be defined. Since a layout of a straight-line representation is completely determined by vertex positions, a very natural notion of stability is the absence of excessive vertex movements. More formally, let the likelihood of y resulting in x consist of vertexwise independent three-dimensional Gaussian distributions with mean at the conditioning position x_v, $v \in \Lambda$,

$$\begin{aligned} P(Y = y \mid X = x) &= \prod_{v \in \Lambda} P(Y_v = y_v \mid X = x) \\ &= \prod_{v \in \Lambda} P(Y_v = y_v \mid X_v = x_v) \\ &= \prod_{v \in \Lambda} \frac{1}{\sigma\sqrt{2\pi}} \, e^{\frac{\|y_v - x_v\|^2}{2\sigma^2}} \end{aligned}$$

$$= \frac{1}{\sigma\sqrt{2\pi}} e^{\sum_{v\in\Lambda} d(y_v,x_v)^2/2\sigma^2}$$

Parameter σ controls the variance of the distribution and therefore the extent to which a vertex is allowed to move. Note that the Gaussian distribution is symmetric in x_v and y_v, and that it immediately translates into an attraction potential. We thus add the local stability criterion

interaction	**potential**
$v \quad : \quad v \in \Lambda$	$\mathsf{Attraction}\,(x_v, y_v \mid c_\sigma)$

with control parameter c_σ to the static layout model from above. This notion of stability is called *anchoring*, because every vertex is attracted by its position in the previous layout. Graphically speaking, stability between layouts is modeled by springs of ideal length zero keeping vertices near a target position derived from the previous layout. This is similar to the approach of Lyons *et al.* (1998) for static graphs with geographic information.

Dynamic layout model. Instead of a single graph with partial preceding layout, the contest data in Figure 3.9 implies a sequence of graphs $G^{(0)}, \ldots, G^{(65)}$, where $G^{(0)}$ is the empty graph. To produce a smooth animation, we now define a dynamic layout model according to which a sequence of layouts is computed. The sequence of positions a vertex takes in these layouts is used as a list of break points for splines defining the trajectory of the vertex during the animation.

For every $G(t)$, $0 < t \leq 65$, three layouts $x(t,0), x(t,1), x(t,2) \in \mathcal{X}(t)$ are computed according to different layout models. The sequence of layouts has the form

$$\begin{array}{llllllll}
x(0) & & & & & & & \\
\downarrow G(1) & & & & & & & \\
x(1,0) & x(1,1) & x(1,2) & & & & & \\
 & & \downarrow G(2) & & & & & \\
 & & x(2,0) & x(2,1) & x(2,2) & & & \\
 & & & & \downarrow G(3) & & & \\
 & & & & x(3,0) & x(3,1) & x(3,2) & \\
 & & & & & & \vdots &
\end{array}$$

where $x(0)$ is empty, $x(t,0)$, $0 < t \leq 65$, is computed from the static model, but with all vertices present in both $G(t-1)$ and $G(t)$ fixed at their position in $x(t-1)$, and $x(t,1)$ and $x(t,2)$ are computed from the static model with the additional stability potential and all vertices movable. That is, each vertex is

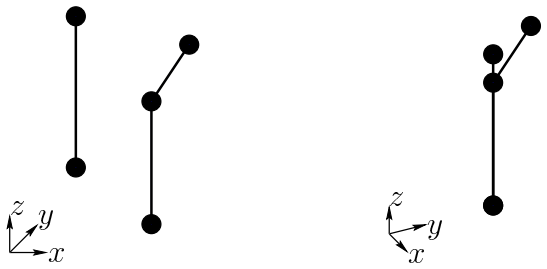


Figure 3.11: Two projections of a three-dimensional layout. All three types of occlusion appear on the right hand side, causing gross misinterpretation of identities and incidences

twice allowed to move into a better position not to distant from its previous one. This way a modification first has no effect on the positions of remaining vertices, but introduces potential new vertices at reasonable positions. A modification of the graph is only then allowed to change the layout on a larger scale. By splitting each update into these three layout steps, we ensure that configurations change gradually. This way 195 layouts are computed to determine the course of the animation. Cubic splines smoothly interpolate vertex positions between these snapshots.

3.3.2 Viewpoints

There are no truly three dimensional media available to display the animation. We therefore seek two-dimensional projections showing as much of the three-dimensional image as possible. Each projection depends on a viewpoint, *i.e.* an observer position together with a direction of view. The selection of good viewpoints under different types of projections and with different measures of goodness is investigated by Kamada and Kawai (1988), Bose *et al.* (1996), and Webber (1997). Here, we use a simple heuristic to find a decent viewpoint under perspective projection once after each modification of the graph, *i.e.* after computing $x(t, 2)$. In the course of the animation we let the observer follow a spline through these points while focusing on the origin.

Starting point for the definition of good viewpoints are bad viewpoints. A viewpoint is said to be bad, if it yields a projection in which an item hides another one, or false incidences are suggested. A projection is said to cause an *occlusion*, if a vertex or point of an edge coincides with another vertex or point of an edge in the projection, but not in the three-dimensional layout. Depending on the type of overlap, we distinguish vertex-vertex, vertex-edge, and edge-edge occlusions, respectively. See Figure 3.11.

For a given distance from the origin, we can imagine the observer sitting on a sphere centered at the origin. Recall that we let the observer focus on the origin at all times. A point on the sphere yields an occlusion, if it is collinear either with two vertices, a vertex and an edge, or two edges. However, edge-edge occlusions are ignored altogether, because the points on the sphere that are collinear with any two points of a pair of non-coplanar edges cover too large an area on the sphere. In general we cannot expect to find any point on the sphere not causing an edge-edge occlusion, anyway.

To further simplify the problem, we subdivide edges with auxiliary vertices and compute only vertex-vertex occlusion points on the sphere. The *goodness* of a viewpoint is then defined to be its minimum great-circle distance from any occlusion point computed. Webber (1997) calls this measure of goodness *rotational separation*. For the animation we approximate the best viewpoint under rotational separation within great-circle distance at most $\pi/2$ from the previous one to suppress large rotations. Finally, another simple heuristic is used to fit the graph onto the screen by adjusting the radius of the observer sphere.

3.3.3 Rendering and Examples

As a small hint on the actual content of the graph, vertices are represented by boxes with corresponding web pages texture-mapped onto their sides. Otherwise the semantics of the graph are not considered in the rendering specification. Figure 3.12 shows the initial frame with manually modified positions to achieve greater effect. Springs are used to render edges, just because they are reminiscent of the physical analogy underlying the layout. It is a by-product that their coils also visualize the stress on an edge. It can be seen from Figure 3.13 that referred pages are hooked to their referrering pages. In the case of bidirectional edges, both ends are hooked. Edges added or deleted in a modification are highlighted during that event. A spherical grid surrounding the graph helps to distinguish between vertex movements and movements of the viewpoint.

Spline break points and rendering information are specified in a POV-RAY[13] scene description file. Using the ray tracer a large number of still frames (50 per modification) is generated. These are subsequently converted into the animation. Figures 3.14 and 3.15 present a number of sample frames from two modification subsequences to give an idea of how the animation proceeds. A more elaborate rendering is shown in Figure 3.16.[14]

[13] See `http://www.povray.org/`.

[14] This image is from a remake of our animation computed on the world record size LINUX cluster put together during the 1998 WDR Computernacht.

Figure 3.12: Web pages are texture mapped onto boxes

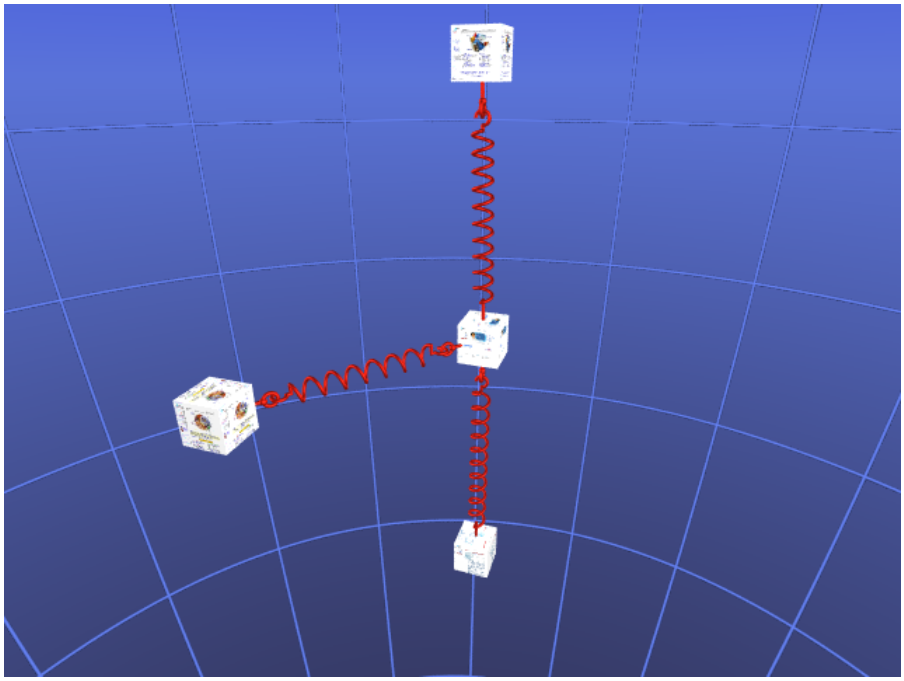


Figure 3.13: Links are represented by springs hooked to the box of the referring page, or both boxes if links are symmetric

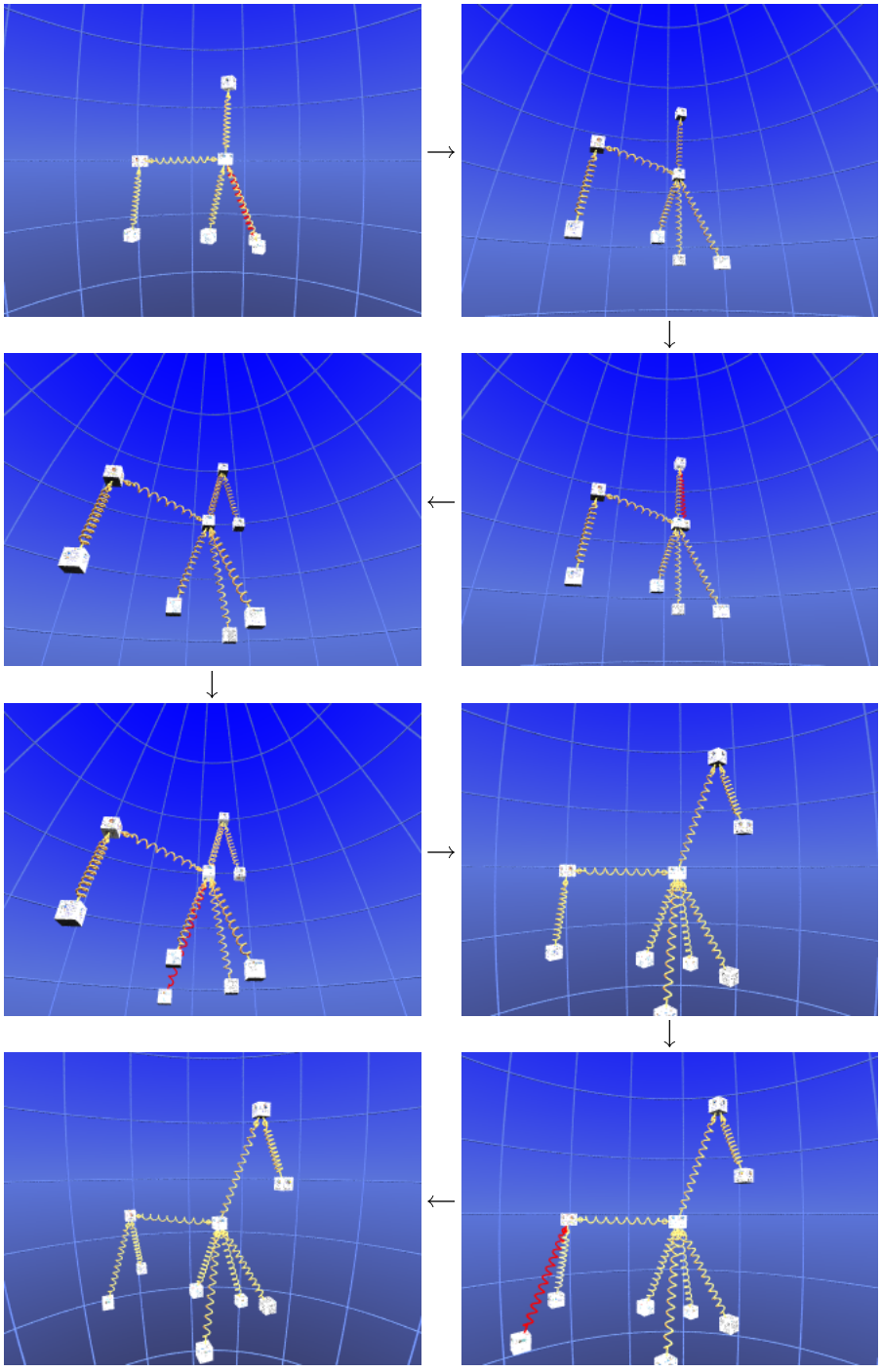


Figure 3.14: First and last frame for modifications six to nine

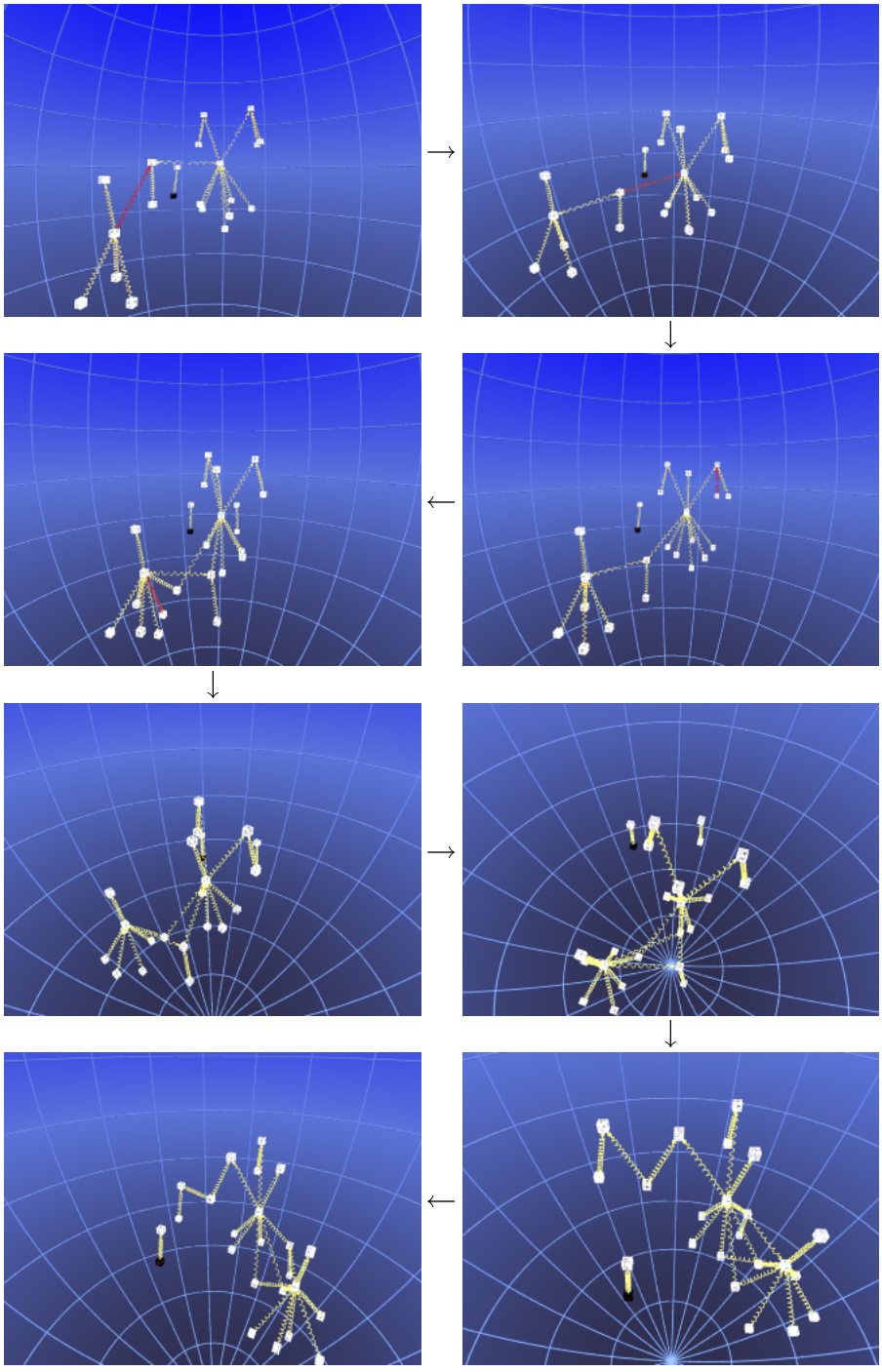


Figure 3.15: Snapshots between modifications 20 and 32

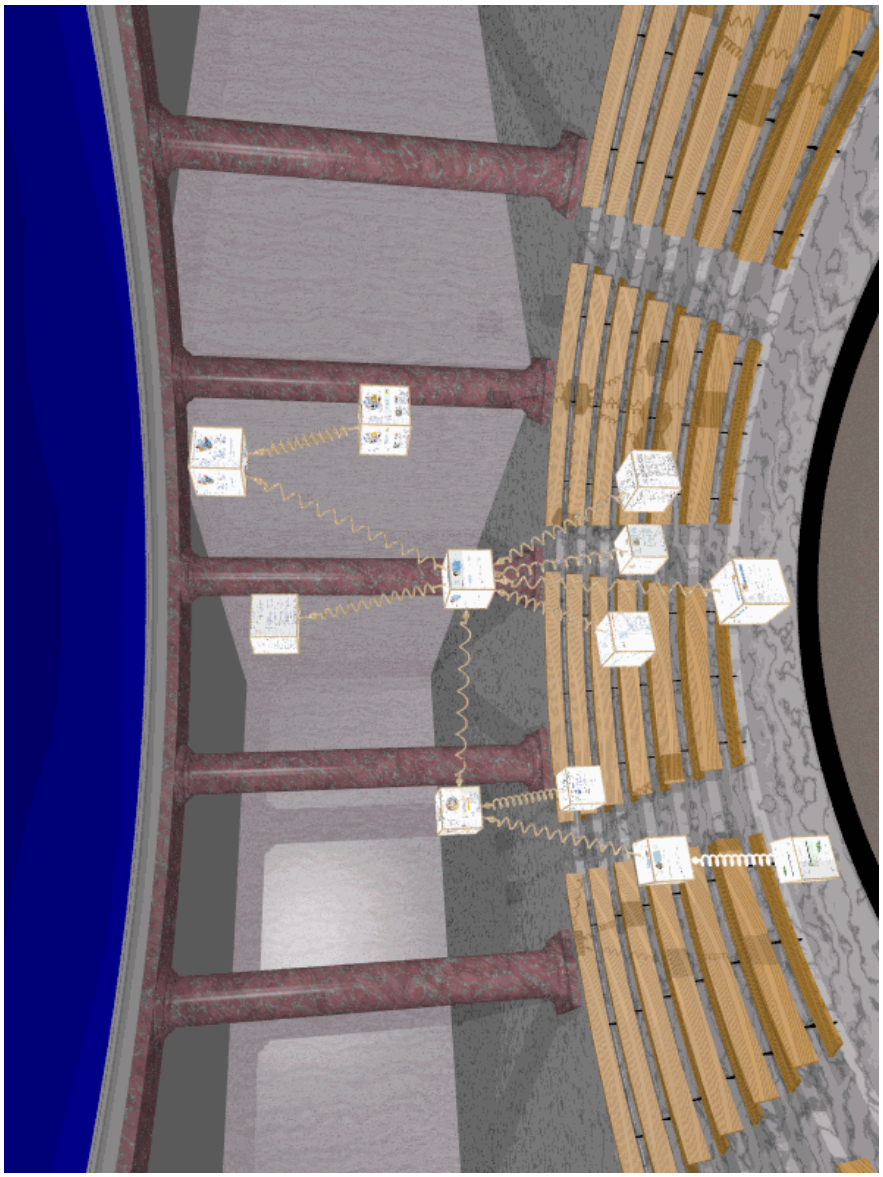


Figure 3.16: The Graph Theater (rendition by Thomas Willhalm)

3.4 Case Study III: Train Connections

The problem of our final case study arises from a cooperation with a subsidiary of the *Deutsche Bahn AG*, *TLC/EVA*. The aim of this cooperation is to develop data reduction and visualization techniques for the explorative analysis of large amounts of time table data from European public transport systems. For the most part these comprise train schedules, but occasionally the data also contains bus, ferry and footpath connections. Analysis of the data with respect to completeness, consistency, changes between consecutive periods of schedule validity, and so on is relevant, *e.g.*, for quality control, (international) coordination, and pricing. Graph layout is to aid visual inspection of this data.

3.4.1 Time Table Graphs

Figures 3.17 and 3.18 show the kind of data that is provided. Since even for moderately important stops like the German part of Konstanz main station there are about 100 trains regularly arriving or leaving, realistic input is quite large. To condense the input, a so-called *time table graph* is built in the following way. For each regular stop of any vehicle, a vertex is inserted into the graph. Two vertices are connected by exactly one edge, if there is any service running from one station to the other (or vice versa) without intermediate stops. Hence, the graphs considered here are simple and undirected.

An important part of the analysis is the classification of edges in two categories: *minimal* edges and *transitive* edges. Minimal edges are those corresponding to a set of continuous connections between two stations not passing through a third one. Typically, these are induced by regional trains stopping at, as they say, every tree. On the other hand, transitive edges correspond to connections passing through other stations without halting. These are induced by through-trains. The substance of a time table graph is the existence or absence of non-stop connections between any two stations, and the classification of each connection into minimal or transitive. Our aim is to design graphical representations clearly displaying the time table graph and a given edge classification.

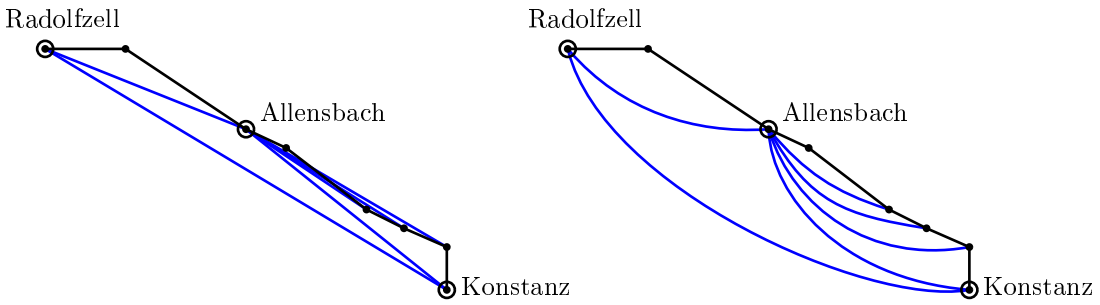
In a graphical design, classification of edges is easily coded using the color property. Figure 3.19(a) shows a small part of a time table graph with edges colored according a given classification. Stations are positioned at their geographical location, and all edges are represented straight-line. An obvious problem for determining the existence or absence of a connection, and for display ergonomics, are edge overlaps and small angles between edges. In order to maintain geographic familiarity, though, we are not allowed to move

```
(...)
8000880 Radolfzell              -58.5  -510.8   (...)
(...)
8003400 Konstanz                -43.5  -519.8   (...)
8003401 Konstanz-Petersh.       -43.5  -518.2   (...)
8003416 Konstanz-Wollmat        -45.1  -517.5   (...)
(...)
8506131 Kreuzlingen             -40.2  -524.5   (...)
(...)
```

Figure 3.17: Excerpts from a station list. Each station has a unique identification number used in the train schedule. Coordinates are in kilometers relative to Hannover (other data omitted)

```
*Z 05130 85    01
*G SE  8506131 8001790
*A VE 8506131 8001790 000000
*A G  8506131 8001790
8506131 Kreuzlingen                 1112  (...)
8003400 Konstanz              1115  1125  (...)
8003401 Konstanz-Petersh.     1127  1128  (...)
8003416 Konstanz-Wollmat      1130  1130  (...)
8004997 Reichenau(Baden)      1132  1133  (...)
8002683 Hegne                 1135  1135  (...)
8000496 Allensbach            1138  1138  (...)
8003872 Markelfingen          1143  1143  (...)
8000880 Radolfzell            1147  1149  (...)
8001059 Böhringen-Rickelsh.   1152  1152  (...)
8000073 Singen(Hohentwiel)    1158  1200  (...)
8004107 Mühlhausen(b Engen)   1206  1206  (...)
8006321 Welschingen-Neuhaus.  1209  1209  (...)
8001790 Engen                 1212        (...)
```

Figure 3.18: Schedule of a single train. It lists all stations hit by the train with arrival and departure time (other data omitted)



(a) straight-line segments (b) Bézier curves

Figure 3.19: Different representations of transitive edges in a small time table graph

vertices, and since minimal edges usually represent actual railways, they are best represented straight-line. It seems therefore reasonable to change the representation of transitive edges to Bézier curves as shown in Figure 3.19(b). These provide the flexibility to route an edge such that overlaps and small angles are resolved. In general, representation of non-stop connections by curved lines not only helps to reduce visual clutter and ambiguity, but also directly resembles the intuition of fast vehicles passing by minor stops.

3.4.2 Curved Edge Layout

The layout elements that need to be positioned to render Bézier curves are their control points. In fact, we may consider stations and control points to be vertices of an auxiliary graph, so that rules for favorable positioning can be modeled by auxiliary edges of appropriate desired length, thus formulating the problem in terms of a straight-line layout. With a different objective in mind, curved edge layout is treated as a routing problem in Dobkin *et al.* (1997) and Gansner and North (1998).

We now give a random field model for the layout of a time table graph $G = (V, E)$. Vertex positions are given by geographical locations of corresponding stations, and we identify vertices with their position. Minimal edges as well as very long transitive edges are represented straight-line.[15] For the other edges we use cubic Bézier curves (cf. Figure 3.20). Let $\breve{E}_{\tau_1} \subseteq E$ be the set of transitive edges of length less than a threshold parameter τ_1, such that

[15]It will be obvious from the examples presented in Section 3.4.3 why it is not useful to represent all transitive edges by Bézier curves.

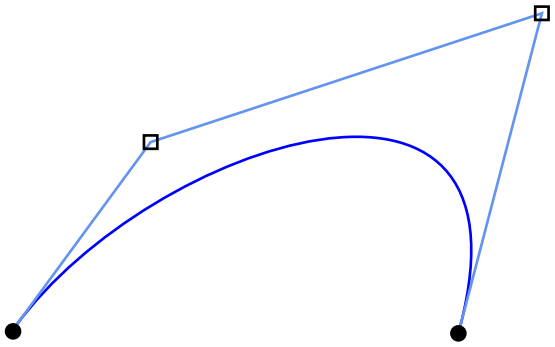


Figure 3.20: Bézier cubic curve (Bézier, 1972). Two endpoints and two control points define a smooth curve that is entirely enclosed by the convex hull of these four points

the set of layout elements consists of two control points for each edge in $\breve{E}_{\tau_1}$, $L = \left\{ b_u(e), b_v(e) \,:\, e = \{u, v\} \in \breve{E}_{\tau_1} \right\}$. If two Bézier points belong to the same edge, they are called *partners*. The *anchor*, $a_{b_v(e)}$, of any $b_v(e) \in L$ is v. All Bézier points are initially positioned on the straight line through the endpoints of their edges at equal distance from their anchor and from their partner.

The position assigned to a Bézier point is influenced by its partner, its anchor, all Bézier points with the same anchor or initially close positions, and all stations near the initial position. Let $\{u, v\} \in \breve{E}_{\tau_1}$ be a transitive edge, and let $b \in L$ be a Bézier point of $\{u, v\}$. Given two parameters ϵ_1 and ϵ_2, consider an ellipse with major axis going through u and v. Let its radii be $\epsilon_1 \cdot \frac{d(u,v)}{2}$ and $\epsilon_2 \cdot \frac{d(u,v)}{2}$, respectively. We denote the set of all stations and Bézier points (at their initial position) within this ellipse, except for b itself, by $\mathcal{E}_b$. Recall that the neighborhood of some layout element consists of all those layout elements that have an influence on its positioning. Therefore, η_b equals the union of $\mathcal{E}_b \cap L$ with the set of Bézier points with the same anchor as b and (since interactions need to be symmetric) the set of Bézier points b' for which $b \in \mathcal{E}_{b'}$. We used $\epsilon_1 = 1.1$ and $\epsilon_2 = 0.5$ for the examples presented in Section 3.4.3.

An interaction potential is defined for each criterion that a good layout of Bézier points should satisfy:

- *Distance to stations.* For each Bézier point $b \in L$ of some edge $\{u, v\} \in \breve{E}_{\tau_1}$, there are repulsion potentials

$$\sum_{w \in \mathcal{E}_b \cap V} \mathsf{Repulsion}\left(x_b, w \;\middle|\; (\varrho_1 \cdot l_b)^4\right)$$

with $l_b = \frac{d(u,v)}{3}$ and ϱ_1 a constant. These ensure reasonable distance from stations in the vicinity of b and can be controlled via ϱ_1. A combined repulsion and attraction potential

$$\mathsf{Distance}\,(x_b, a_b \mid l_1 \cdot l_b\,)$$

with l_1 a distance controlling constant, keeps b sufficiently close to its anchor a_b.

- *Distance to near Bézier points.* As is the case with near stations, a Bézier point $b_1 \in L$ should not lie too close to another Bézier point $b_2 \in \eta_{b_1}$. If b_1 is neither the partner of nor bound to b_2 (binding is defined below), we add

$$\mathsf{Repulsion}\left(x_{b_1}, x_{b_2} \;\middle|\; \varrho_2^4 \cdot \min\{l_{b_1}^4, l_{b_2}^4\}\right)$$

The desired distance between partners b_1 and b_2 is equal to the desired distance from their respective anchors,

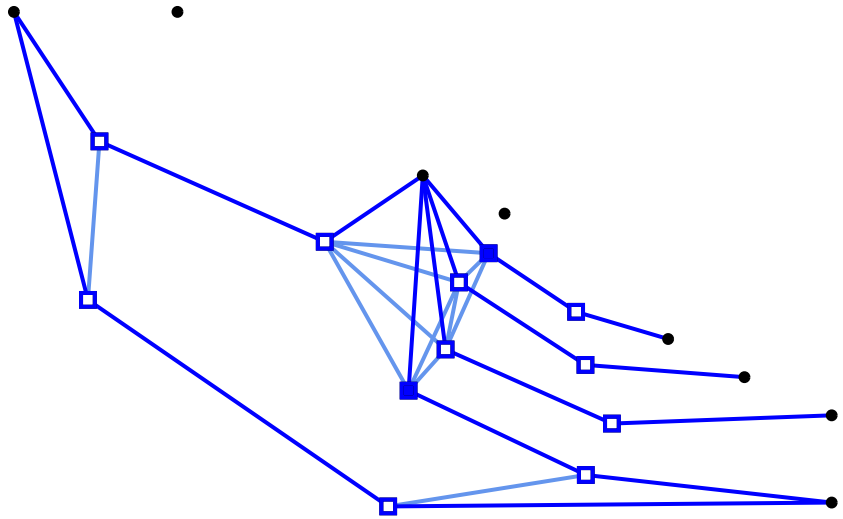
$$\mathsf{Distance}\,(x_{b_1}, x_{b_2} \mid l_1 \cdot l_{b_1}\,)$$

- *Binding.* In general, it is not desirable to have Bézier points $b_1, b_2 \in L$ with a common anchor lie on different sides of a minimal edge path through the anchor. Therefore, we *bind* them together, if l_{b_1} does not differ much from l_{b_2}, *i.e.* if $\frac{1}{\tau_2} < \frac{l_{b_1}}{l_{b_2}} < \tau_2$ for a threshold $\tau_2 \geq 1$, we add potentials

$$\omega \cdot \mathsf{Distance}\,(x_{b_1}, x_{b_2} \mid l_2 \cdot (l_{b_1} + l_{b_2})/2\,)$$

where l_2 is a stretch factor for the length of binding edges, and ω controls the importance of binding relative to the other potentials.

Altogether, this following layout model is made of nothing but attraction and repulsion potentials that define an auxiliary graph layout problem in the following way: Stations correspond to vertices with fixed positions, while Bézier points correspond to vertices to be positioned. Edges of different desired lengths exist between Bézier points and their anchors, between partners, and between Bézier points bound together. Just like edge lengths, the magnitude of repulsion differs across the elements. See Figure 3.21 and recall that repulsion potentials are defined on local neighborhoods. The respective influence of different parameters is discussed in the following section.



interaction			potential
$b_e(v)$	:	$b_e(v) \in L$	$\sum_{u \in \mathcal{E}_{b_e(v)} \cap V} \mathsf{Repulsion}\left(x_{b_e(v)}, u \;\middle\vert\; (\varrho_1 \cdot l_{b_e(v)})^4\right)$
$b_e(v)$	:	$b_e(v) \in L$	$\mathsf{Distance}\left(x_{b_e(v)}, v \;\middle\vert\; l_1 \cdot l_{b_e(v)}\right)$
$b_e(v), b_{e'}(v')$	:	$v \neq v', e \neq e', b_{e'}(v') \in \mathcal{E}_{b_e(v)}$	$\mathsf{Repulsion}\left(x_{b_e(v)}, x_{b_{e'}(v')} \;\middle\vert\; \varrho_2^4 \cdot \min\{l_{b_e(v)}^4, l_{b_{e'}(v')}^4\}\right)$
$b_e(v), b_{e'}(v')$	:	$v \neq v', e = e'$	$\mathsf{Distance}\left(x_{b_1}, x_{b_{e'}(v')} \;\middle\vert\; l_1 \cdot l_{b_e(v)}\right)$
$b_e(v), b_{e'}(v')$	:	$v = v', e \neq e', \frac{1}{\tau_2} < \frac{l_{b_e(v)}}{l_{b_{e'}(v')}} < \tau_2$	$\omega \cdot \mathsf{Distance}\left(x_{b_e(v)}, x_{b_{e'}(v')} \;\middle\vert\; l_2 \cdot (l_{b_e(v)} + l_{b_{e'}(v')})/2\right)$

Figure 3.21: Time table graph layout model. For illustration, the auxiliary graph induced by Bézier point layout interactions for the time table graph of Figure 3.19(b) is shown. Note that there is no binding between the two layout elements indicated by filled rectangles, because anchor distances are too different (threshold parameter τ_2)

3.4.3 Experiments

The objective function described in the previous section was obtained only after experimentation with a number of different potentials and parameters. We started with a simple combination of repulsion from stations, and attraction and repulsion from partners and anchors. In fact, we then used splines to represent transitive edges. It seemed that they offered better control, since they actually pass through their control points. However, segments between partners tended to extend far into the layout area. After replacing splines by Bézier curves, the promising results encouraged us to try more elaborate objective functions. It turned out that it is useful to represent long transitive edges straight-line, which led to the introduction of threshold τ_1. A new requirement we found while looking at earlier examples was that incident (consecutive or nested) transitive edges should lie on one side of a path of minimal edges. Binding proved to achieve this goal, but needed to be constrained to control segments of similar desired length, because otherwise short transitive edges are deformed when bound to long ones. Threshold τ_2 therefore controls the length ratio of bound segments.

The identification of a suitable vector $\theta = (\varrho_1, \varrho_2, l_1, l_2, \omega, \tau_1, \tau_2)$ of parameters is a serious problem. Mendonça and Eades (1993) use two nested simulated annealing computations to identify parameters of an energy based placement model. In Masui (1994), a genetic algorithm is used to breed a suitable objective function. However, both methods are heuristic in defining their objective as well as in optimizing it. Given one or more examples which are considered to be well done (*e.g.* by manual rearrangement), a theoretically sound approach would be to carry out parameter estimation for random variable $X(\theta)$ describing the layout model as a function of parameter vector θ. Given a layout x, the likelihood of θ is

$$P(X = x \,|\, \theta) = \frac{1}{Z(\theta)} \exp\{-U(x \,|\, \theta)\}$$

where $Z(\theta) = \sum_{y \in \mathcal{X}} \exp\{-U(y \,|\, \theta)\}$ is the normalizing constant. A maximum likelihood estimate θ^* is obtained by maximizing the above expression with respect to θ. Unfortunately, computation of $Z(\theta)$ is practically intractable, since it sums over all possible layouts. One might hope to reduce computational demand by exploiting the locality of random fields (see, *e.g.*, Winkler, 1995). Even though neighboring layout elements are clearly not independent, reasonable estimates are obtained from the pseudo-likelihood function (Besag, 1986)

$$\prod_{\lambda \in L} \frac{1}{Z_\lambda(\theta)} \exp\left\{ - \sum_{C \in \mathcal{C} \,:\, \lambda \in C} U_C(x \,|\, \theta) \right\}$$

with $Z_\lambda(\theta) = \sum_{x_\lambda \in \mathcal{X}_\lambda} \exp\{-\sum_{C \in \mathcal{C}\,:\,\lambda \in C} U_C(x \mid \theta)\}$. However, $Z_\lambda(\theta)$ sums over all possible positions of layout element λ, such that maximization is still intractable in this setting. So we exploit locality in a very different way, namely by experimenting with small examples in a feedback cycle. The parameters θ thus identified proved appropriate even for huge graphs, because the model scales so well.

Figure 3.22 lists the rationale behind each parameter in $\theta = (\varrho_1, \varrho_2, l_1, l_2, \omega, \tau_1, \tau_2)$ together with our final choice of values. The effects of some parameters are demonstrated in Figure 3.23. Figures 3.23(a) and 3.23(b) show how increased repulsion parameters spread Bézier points. Without binding, curves tend to lie on different sides of minimal edges as in Figure 3.23(c), and Figure 3.23(d) shows that this can even be enforced. Obviously, binding is a valuable refinement.

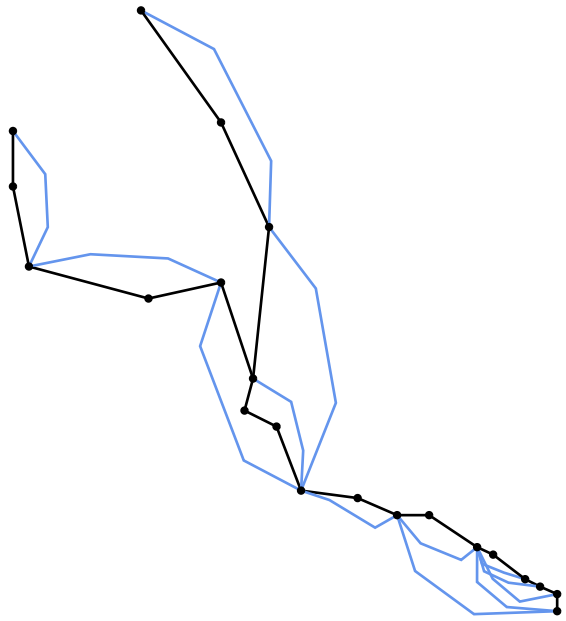
Our implementation of a random field layout module (Section 2.2.3) enabled us to experiment with parameters in all of the above layout models without implementing each preliminary model from scratch.

The original data sets provided by *TLC/EVA* are quite large: For a time table graph of the size shown in Figure 3.27 (roughly 2,000 vertices and 4,000 edges), about 11 MByte of time table data were evaluated. Connections are classified into minimal and transitive edges using existing code.

A first example is shown in Figure 3.24. The graph contains regional trains in south-west Germany. Edge classification, transformation into a layout graph, neighborhood generation, and layout computation took less than two minutes. The example demonstrates how visual inspection can immediately yield some candidates for misclassified edges. Parts of the drawing are magnified in Figures 3.25 and 3.26. A few labels have been added to support geographical location of the area shown, but otherwise the drawings have not been modified. Verify that connections can be told apart quite well, and that binding successfully causes incident (consecutive or nested) transitive edges to lie on the same side of minimal edges.

Larger examples are given in Figures 3.27 and 3.29. One readily observes that the algorithm scales very well, *i.e.* increased size of the graph does not reduce layout quality on more detailed levels as can be seen in Figures 3.28 and 3.4.3. This is largely due to the fact that neighborhoods remain fairly local. Together with the ability to zoom into different regions, data exploration is well supported. The advantage of a length threshold for curved transitive edges is another straightforward observation, notably in Figures 3.29 and 3.30(a).

On these larger examples, the implementation based on simulated annealing is rather slow. But since our final model uses only attraction and repulsion potentials, the potentials can be replaced by equivalent attractive

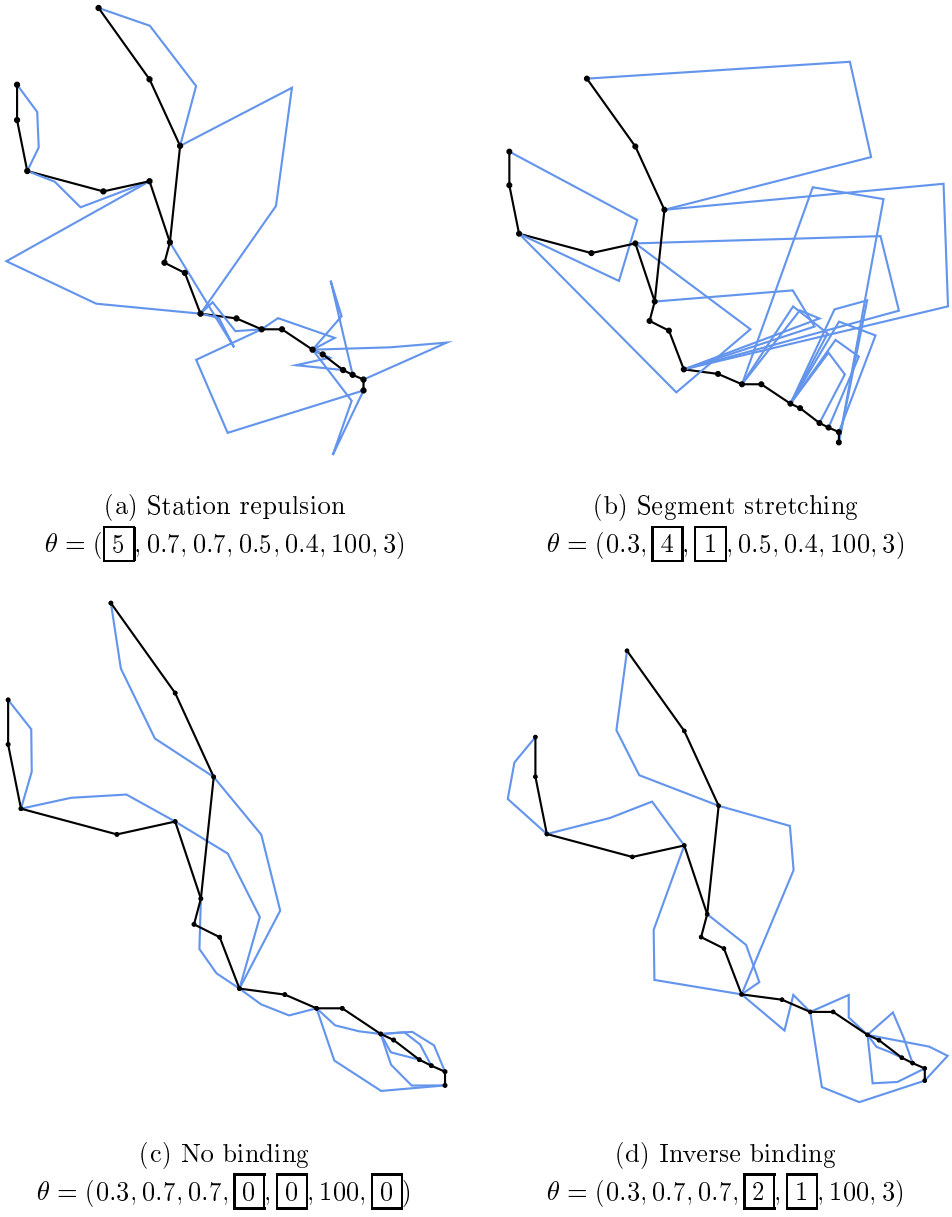


(a) $\theta = (0.3, 0.7, 0.7, 0.5, 0.4, 100, 2.2)$

	controls
ϱ_1	distance of Bézier points from stations
ϱ_2	mutual distance of Bézier points
λ_1	length of control segments
λ_2	length of bands
ω	importance of binding
τ_1	threshold for straight transitive edges
τ_2	threshold for binding segments of different length
ϵ_1	major axis radius of neighborhood defining ellipse
ϵ_2	minor axis radius of neighborhood defining ellipse

(b) Parameters of the time table graph layout model

Figure 3.22: Parameters in the time table graph layout model and a recommended choice applied to a small time table graph. Control segments shown instead of Bézier curves (cf. Figure 3.23)



(a) Station repulsion
$\theta = (\boxed{5}, 0.7, 0.7, 0.5, 0.4, 100, 3)$

(b) Segment stretching
$\theta = (0.3, \boxed{4}, \boxed{1}, 0.5, 0.4, 100, 3)$

(c) No binding
$\theta = (0.3, 0.7, 0.7, \boxed{0}, \boxed{0}, 100, \boxed{0})$

(d) Inverse binding
$\theta = (0.3, 0.7, 0.7, \boxed{2}, \boxed{1}, 100, 3)$

Figure 3.23: Effects of some parameters demonstrated. For ease of comparison, control segments are shown instead of the corresponding Bézier curves. All examples have $\epsilon_1 = 1.1$ and $\epsilon_2 = 0.5$ and should be compared to Figure 3.22

and repulsive forces which are much more amenable to deterministic local optimization. By implementing an algorithm similar to that of Fruchterman and Reingold (1991), layout computation times could be reduced by a factor of ten to within a few minutes.

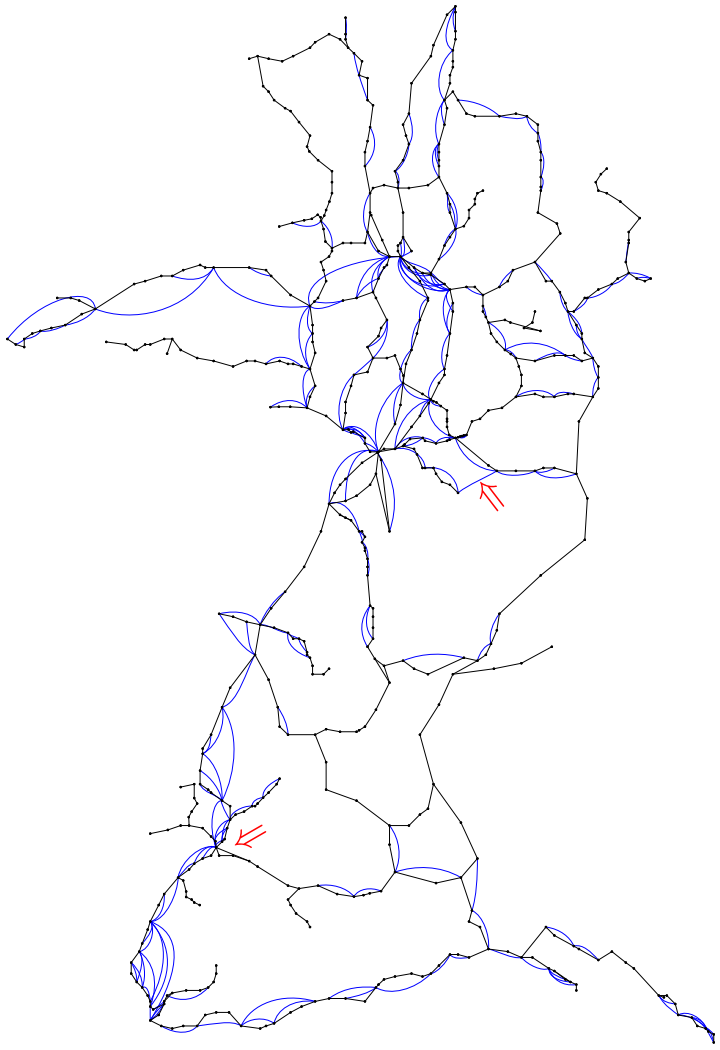


Figure 3.24: Regional trains in south-west Germany. 619 vertices, 876 edges (229 transitive), $\theta = (0.7, 0.3, 0.7, 0.5, 0.4, 100, 3)$. Arrows point out two out of several edges that appear to be misclassified

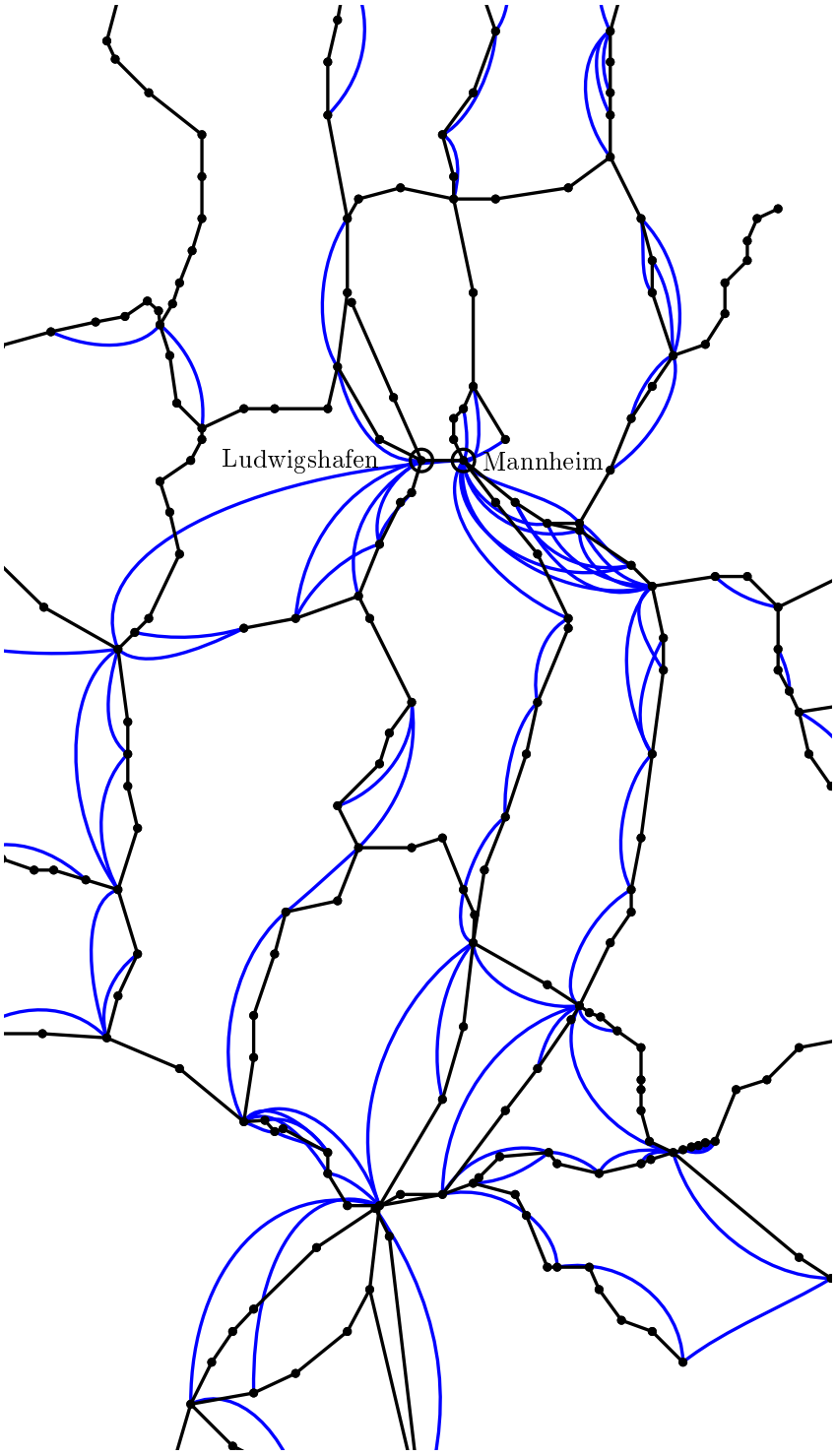


Figure 3.25: Magnification from Figure 3.24

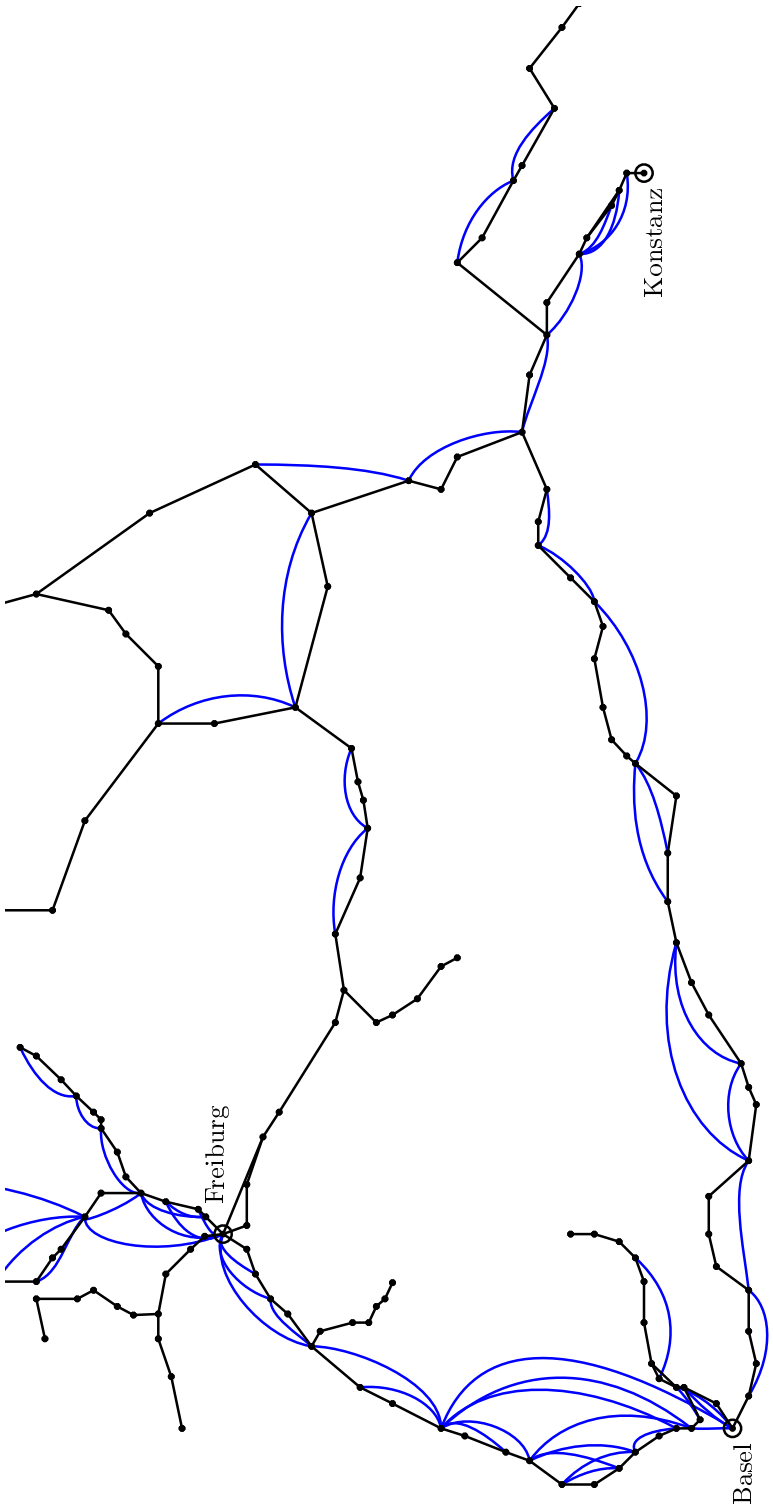


Figure 3.26: Magnification from Figure 3.24

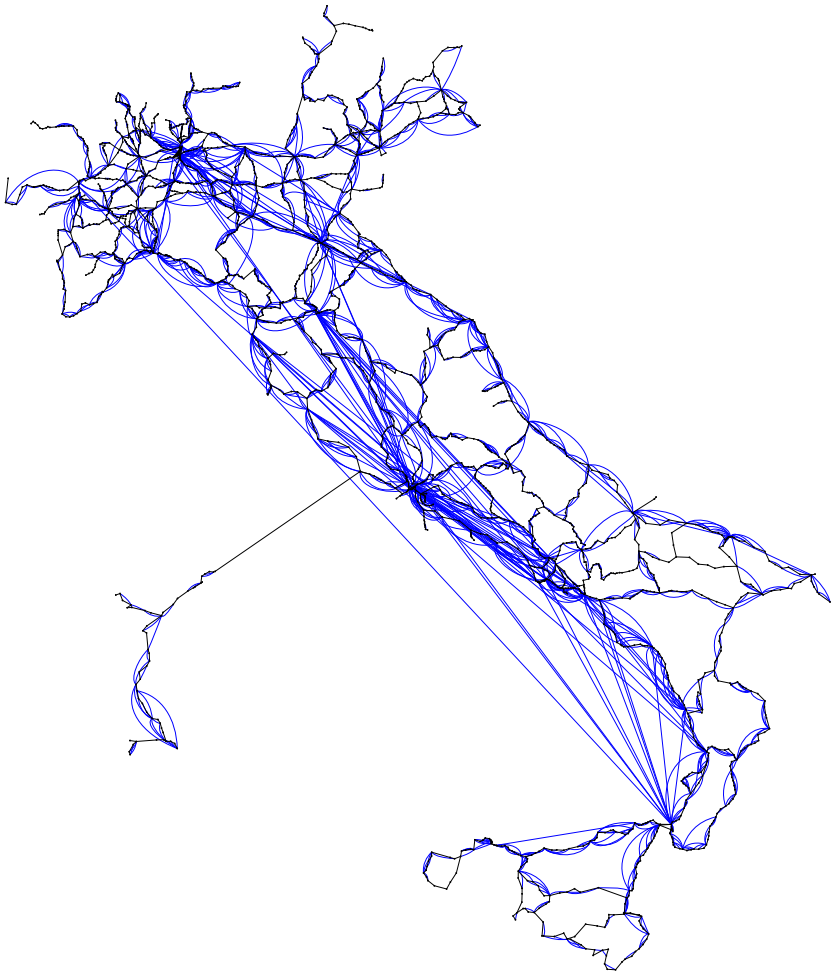


Figure 3.27: Italian train and ferry connections. 2,386 vertices, 4,370 edges (1,849 transitive), $\theta = (0.7, 0.3, 0.7, 0.5, 0.4, 100, 3)$

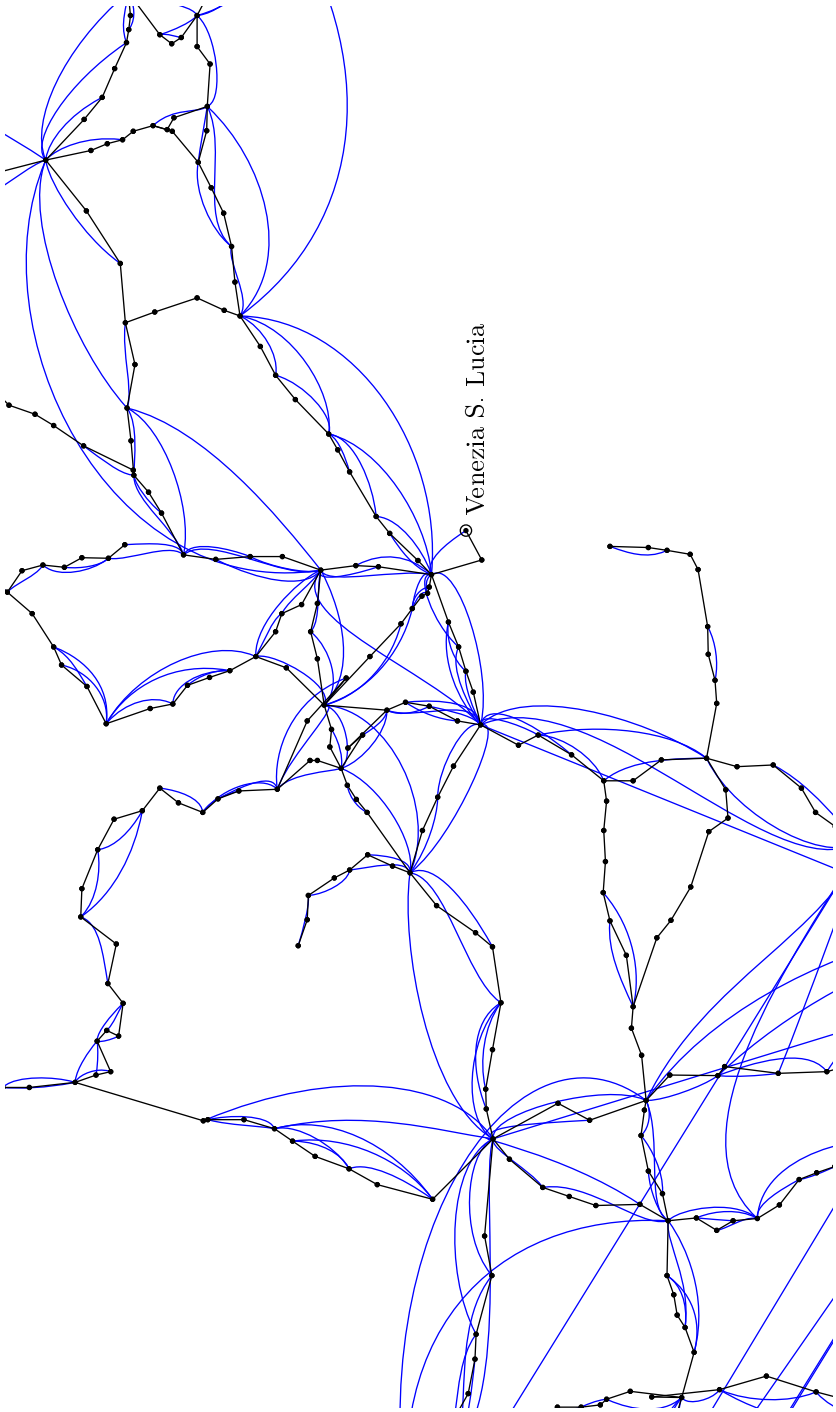


Figure 3.28: Magnification from Figure 3.27

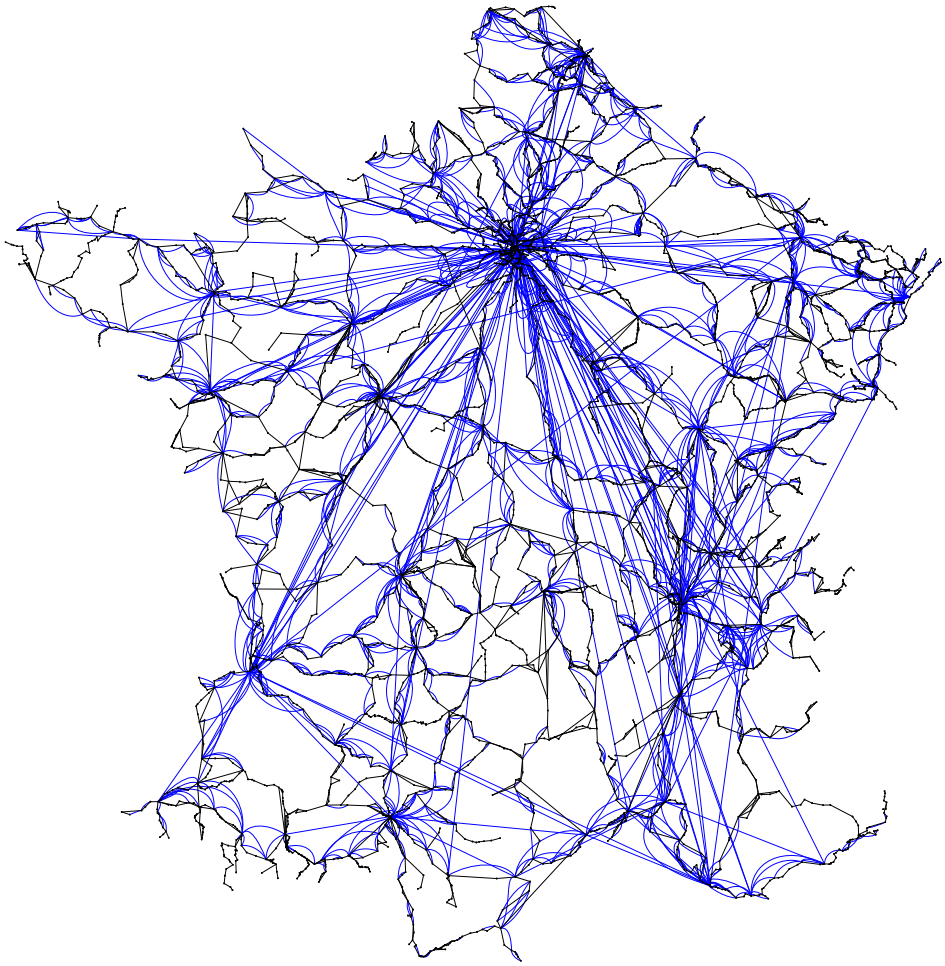
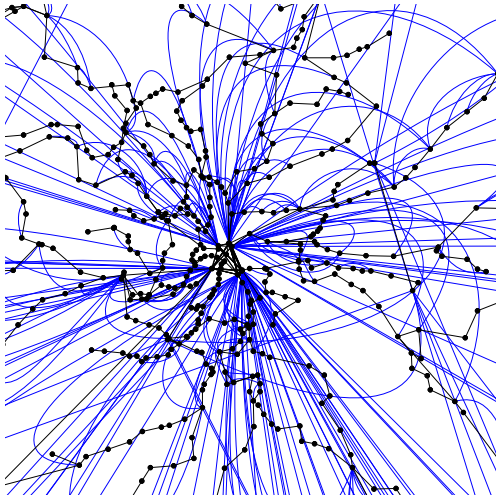
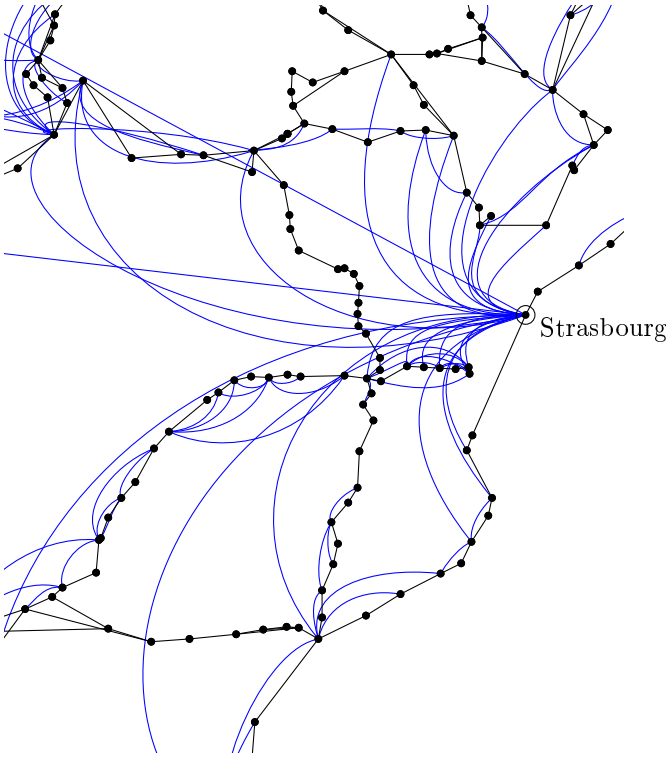


Figure 3.29: French connections. 4,551 vertices, 7,793 edges (2,408 transitive), $\theta = (0.7, 0.3, 0.7, 0.5, 0.4, 100, 3)$



(a) Paris has six long-distance stations



(b) Strasbourg, gateway to France

Figure 3.30: Magnifications from Figure 3.29

Chapter 4

Intermezzo

The case studies of the previous chapter demonstrated that energy based placement, together with a general (approximate) optimization scheme, lends itself to rapid prototyping of layout models. This is because of the inherent flexibility to arbitrarily combine various local criteria. However, when a satisfactory model is identified, such flexibility is an unnecessary feature dominated by the need for faster layout computation.

A closer look at the time table graph layout model of Section 3.4 revealed that a faster, deterministic, algorithm could be applied to obtain approximate solutions of the same quality. In this chapter, we discuss a family of energy based placement models which, despite their simplicity, have a number of very interesting properties. In particular, it turns out that optimal layouts can be determined efficiently. Recall that energy minimization was shown to be $\mathcal{NP}$-hard in general (Theorem 2.1).

However, it turns out that these models have some provable drawbacks, too. One of them is that angles formed by incident edges may be tiny, a problem that was circumvented by curved representation of transitive edges in time table graphs. This chapter thus bridges the gap to the next, in which layout methods are considered that determine angles in graphical presentations prior to positions of graphical marks.

4.1 Barycentric Layouts

A surprisingly interesting layout model for straight-line representations of simple, undirected graphs $G = (V, E)$ in d-dimensional Euclidean space is the simplistic *barycentric layout model* (Tutte, 1963)

interaction	**potential**
$u, v \quad : \quad \{u, v\} \in E$	$\mathsf{Attraction}\,(x_u, x_v)$

Similar to the spring embedder, this layout model can be interpreted in terms of a spring system. Due to the absence of repulsion the ideal length of the springs is zero, though. Since different connected components do not interact, we restrict ourselves to connected graphs throughout this section.

Obviously, every layout assigning the same position to all vertices is optimal, and every optimal layout has this rather useless form. So, what could possibly be interesting about this model?

A natural way to explore an imaginary spring system of the above type would be to drag some some of the vertices apart and watch the structure unfold. More formally, the model is generalized by introducing fixed positions $y = (y_v)_{v \in V_0}$ for some subset $V_0 \subseteq V$ of vertices. The barycentric layout model is therefore considered in the following form

interaction	**potential**
$u, v \quad : \quad \{u, v\} \in E$	$\mathsf{Attraction}\,(x_u, x_v)$
dependency	**constraint**
$v \quad : \quad v \in V_0$	$x_v = y_v$

Three naturally arising questions are treated in this section:

1. What do optimal layouts look like?
2. Can they be efficiently determined?
3. Why is this model called barycentric, anyway?

4.1.1 Optimal Layouts

The energy of a d-dimensional layout in the barycentric model is

$$U(x) \;=\; \sum_{\{u,v\} \in E} d(x_u, x_v)^2 = \sum_{i=1}^{d} \sum_{\{u,v\} \in E} \left(x_u^{(i)} - x_v^{(i)}\right)^2$$

where $x_v = \langle x_v^{(1)}, \ldots, x_v^{(d)} \rangle$, $v \in V$. A necessary condition for layouts to minimize this energy is that partial derivatives vanish, that is

$$\frac{\partial U}{\partial x_v^{(i)}} \quad = \sum_{u \,:\, \{u,v\} \in E} 2\left(x_v^{(i)} - x_u^{(i)}\right) = 0$$

for all $v \in V \setminus V_0$, $i = 1, \ldots, d$. Obviously, coordinates can be considered independently. For convenience we hence state the subsequent observations in terms of one-dimensional layouts, and it is understood that this layout may

represent but one dimension of a higher dimensional problem. Rewriting the necessary conditions for all $v \in V \setminus V_0$ into

$$x_v = \frac{1}{d_G(v)} \sum_{u\colon \{u,v\} \in E} x_u$$

shows that, in an optimal layout, each vertex of $V \setminus V_0$ is positioned in the barycenter of its neighbors. We have thus answered the question about the name of the model, and we have also found a first property of optimal layouts.

For ease of notation, henceforth assume that $V = \{1, \ldots, n\}$ and $V_0 = \{v : v \leq k\}$ for some $k \in \{0, \ldots, n\}$. Let $A(G)$ be the adjacency matrix of G with rows and columns ordered according to vertex numbers, and define the diagonal *degree matrix* $D(G) = (d_{uv})_{1 \leq u,v \leq n}$ of G by

$$d_{uv} = \begin{cases} d_G(v) & \text{if } u = v \\ 0 & \text{otherwise} \end{cases}$$

The above optimality conditions can be stated more compactly using the *Laplacian matrix* $L(G) = (l_{uv})_{u,v \in V} = D(G) - A(G)$. If $V_0 = \emptyset$, *i.e.* no vertex position is fixed in advance, optimal layouts satisfy

$$L(G) \cdot x = \mathbf{0}$$

If $V_0 \neq \emptyset$, some equations are not required since fixed vertex positions need not satisfy the local optimality conditions. Let $L(G)^{uv}$ be the matrix obtained from $L(G)$ by striking out the u-th row and v-th column, where $L(G)^{uv,u'v'}$ is shorthand for $(L(G)^{uv})^{u'v'}$. Note that rows and columns can be dropped in any order. We hence let $L(G)^{V'}$ denote the matrix obtained from $L(G)$ by striking out the rows and columns of all vertices of some $V' \subseteq V$. The generalized form of the optimality conditions is given by the smaller system in which prescribed positions are brought to the right hand side

$$L(G)^{V_0} \cdot x_{V \setminus V_0} = \left(\sum_{u \in V_0 : \{u,v\} \in E} y_u \right)^{\mathsf{T}}_{v \in V \setminus V_0}$$

To proof that this system of linear equations is indeed solvable, and moreover, that it has a unique solution, it is shown that $|L(G)^{V_0}|$, the determinant of $L(G)^{V_0}$, is positive.

First note that for the graph G/V_0 obtained from G by contracting the vertices of V_0 into a new vertex v_0 and omitting all resulting loops, $L(G/V_0)^{v_0 v_0} = L(G)^{V_0}$. That is, the same matrix is encountered in a barycentric layout model that fixes only a single vertex position.

The following beautiful theorem is due to Kirchhoff (1847), and its use for graph layout was already noted in Maxwell (1874). A commonly cited proof is from Trent (1954), but relies on the Binet-Cauchy formula for determinants of matrix products. We here state a compact version of the elementary proof of Brooks *et al.* (1940) and, independently, Hutschenreuther (1967). A number of generalizations is given in Chaiken and Kleitman (1978).

A graph $G' = (V', E')$ is a *subgraph* of a graph $G = (V, E)$, if $V' \subseteq V$ and $E' \subseteq E$. It is a *spanning* subgraph, if $V' = V$. A graph is called a *tree*, if it is connected and contains no cycles. A spanning subgraph that is a tree is called a *spanning tree*.

Theorem 4.1 (Matrix Tree Theorem) *For every vertex v of a multigraph G, $|L(G)^{vv}|$ equals the number of spanning trees of G.*

■ **Proof** Let $t(G)$ denote the number of spanning trees of G. If G contains only a single vertex v, then $|L(G)^{vv}| = t(G) = 1$ by definition, and if G is not connected, $|L(G)^{vv}| = t(G) = 0$ for every $v \in V$. Therefore, let G be a connected graph with more than one vertex. For every vertex $v \in V$ there is at least one edge $e = \{u, v\} \in E$ that is not a loop. Let $G - e$ denote the graph obtained by deleting the edge e, and G/e denote the graph obtained by contracting u and v (and all edges between them) into a new vertex v_e. All spanning trees of G either contain e, or do not contain e. Hence, $t(G) = t(G/e) + t(G - e)$. Moreover, expanding by the row of u,

$$\begin{aligned}
|L(G)^{vv}| &= \sum_{w \neq v} l_{uw} \cdot \sigma_v(u,w) \cdot |L(G)^{vv,uw}| \\
&= \sum_{w \neq u,v} l_{uw} \cdot \sigma_v(u,w) \cdot |L(G)^{vv,uw}| \\
&\qquad +(l_{uu} - 1) \cdot |L(G)^{vv,uu}| + |L(G)^{vv,uu}| \\
&= \sum_{w \neq u,v} l_{uw} \cdot \sigma_v(u,w) \cdot |L(G)^{vv,uw}| \\
&\qquad +(d_G(u) - 1) \cdot \sigma_v(u,u) \cdot |L(G)^{vv,uu}| + |L(G/e)^{v_e v_e}| \\
&= |L(G-e)^{vv}| + |L(G/e)^{v_e v_e}|
\end{aligned}$$

where

$$\sigma_v(u,w) = \begin{cases} (-1)^{u+w} & \text{if } v < u, w \text{ or } v > u, w \\ (-1)^{u+w-1} & \text{if } u < v < w \text{ or } w < v < u \end{cases}$$

The theorem follows from induction on the number of vertices and edges of G. □

Since G/V_0 is connected if G is, there is at least one spanning tree and therefore $|L(G/V_0)^{v_0v_0}| = |L(G)^{V_0}| > 0$. Non-negativity of the barycentric energy then implies that there is a unique layout satisfying the optimality conditions.

Corollary 4.2 *Let $G = (V, E)$ be a connected graph and $V_0 \subseteq V$ any non-empty set of vertices with fixed positions. Then there is a unique optimal layout in the barycentric layout model which can be determined by solving, for each dimension of the layout space, a system of $|V \setminus V_0|$ linear equations.*

Standard methods for solving systems of linear equations hence lead to an $\mathcal{O}(n^3)$ algorithm for energy minimization in the barycentric layout model. If the input graph is sparse, more advanced methods can be applied.

The following important class of sparse graphs will play a crucial role in the next chapter. A two-dimensional representation mapping vertices to points and edges to continuous curves is called *planar*, if no to curves intersect except at their endpoints. A graph is called *planar*, if it has a planar representation. The cyclic orderings of edges around vertices in any planar representation form a planar (combinatorial) *embedding* of the graph. Interestingly, every planar embedding of a simple planar graph can be realized in a planar straight-line representation (Wagner, 1936; Fáry, 1948; Stein, 1951).

Since there are no edge crossings in planar representations, they subdivide the layout area into connected regions. Each cyclic list of edges encountered when walking around one such region, and sometimes the region itself, is called a *face* of the embedded graph. Note that each edge is encountered exactly twice, possibly in the same list. The number of edges encountered around a face f is called the degree of the face and denoted by $d_G(f)$. A planar embedded graph is therefore given by a triple $G = (V, E, F)$, where F denotes the set of faces. To limit the set of possible planar representations one face might be prescribed to be the *external* (*i.e.* unbounded) face. All other faces are then called *internal*.

For planar graphs, the system of linear equations of the barycentric layout model can be solved in time $\mathcal{O}(n^{1.5})$ (Lipton *et al.*, 1979), but some planar graphs reveal even more surprising properties of the barycentric layout model. Any graph is *biconnected* (*triconnected*), if at least two (three) vertices must be removed to disconnect the graph. Triconnected planar graphs are especially interesting, *e.g.* because they have only one planar embedding and because a triconnected graph is planar, if and only if it is the skeleton of a convex polyhedron (Steinitz and Rademacher, 1934).

Theorem 4.3 (Tutte 1963) *Let G be a planar triconnected graph and V_0 be the set of vertices incident to the edges of a designated external face f_0*

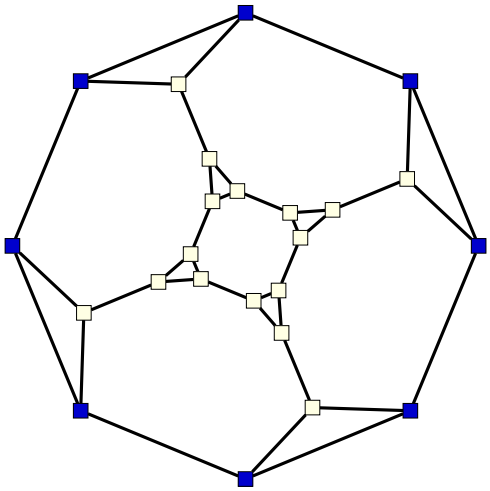


Figure 4.1: An optimal layout of a triconnected planar graph (*cube connected cycles*) in the barycentric layout model. Vertices with prescribed position have darker color

in the unique planar embedding of G. If $(y_v)_{v \in V_0}$ maps this face to a convex polygon preserving the cyclic order of vertices, then the unique optimal layout of the barycentric layout model is planar, and every face is mapped to a convex polygon.

It is not necessary that the input graph be triconnected. For example, vertices subdividing edges and separation pairs contained in V_0 do not affect the layout properties claimed. Thomassen (1980) characterizes the class of planar graphs that admit any planar straight-line representation with convex faces.

Unfortunately, optimal barycentric layouts in general display some less desirable properties, too. The next section discusses two of them.

4.1.2 Drawbacks

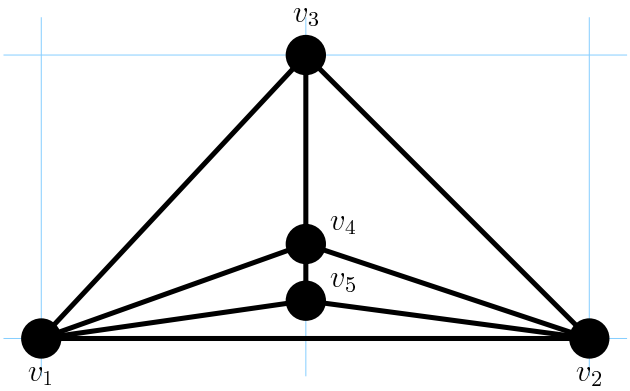
The barycentric layout model is conveniently described by a system of linear equations with integer coefficients. An immediate consequence is that, if the positions assigned to vertices in $V_0 \neq \emptyset$ have rational coordinates, then all coordinates in the unique optimal layout are rational. By multiplication with a constant they can be made integer, such that vertices of an optimal layout can be placed on a sufficiently large integer grid. However, the following result of Eades and Garvan (1996) shows that this grid has exponential area in the worst case. That is, barycentric layouts may require exceedingly high display

resolution and produce non-uniform edge lengths. We present a slightly modified version of both the original theorem and proof.

Resolution of straight-line representations is defined in Eades and Garvan (1996) as the ratio of the smallest distance between vertex positions to the diameter of the set of vertex positions. We propose a slightly different notion capturing the case that all vertices do have sufficient distance, but some are only slightly off along one coordinate axis. Therefore, the *(spatial) resolution* of a layout is defined to be the smallest non-zero ratio of the smallest to largest difference in one coordinate between two vertex positions. Note that this definition allows for different scalings along coordinate axes.

Theorem 4.4 (Eades and Garvan 1996) *The worst case resolution of optimal layouts in the barycentric layout model is* $\mathcal{O}(\frac{1}{l^n})$, $l > 1$.

■ **Proof** Consider the family of triconnected graphs $G(3), G(4), \ldots$, where $G(3)$ is a triangle with vertices v_1, v_2, v_3, and $G(n)$, $n > 3$, is obtained from $G(n-1)$ by introducing a new vertex v_n and connecting it to v_1, v_2, and v_{n-1}. For each of these graphs, positions are prescribed for vertices v_1, v_2, and v_3, namely $y_{v_1}(n) = \langle -1, 0\rangle$, $y_{v_2}(n) = \langle 1, 0\rangle$, and $y_{v_2}(n) = \langle 0, 1\rangle$. For $n = 5$ we have



Let $x(3), x(4), \ldots$ be the sequence of optimal layouts in the barycentric layout model. The y-coordinates satisfy $x^{(2)}_{v_{n-1}}(n) < x^{(2)}_{v_{n-1}}(n-1)$ for all $n > 4$, and $x^{(2)}_{v_n}(n) = \frac{1}{3} \cdot \left(x^{(2)}_{v_1}(n) + x^{(2)}_{v_2}(n) + x^{(2)}_{v_{n-1}}(n)\right) = \frac{1}{3} \cdot x^{(2)}_{v_{n-1}}(n)$. The resolution of $x(n)$, $n > 3$ is hence at most $x^{(2)}_{v_n}(n)/x^{(2)}_{v_3}(n) \leq \frac{1}{3^{n-3}}$. The argument generalizes to barycentric layouts in any other number of dimensions. □

Exponentially small worst case resolution is poor compared to the optimal $\mathcal{O}(\frac{1}{\sqrt[d]{n}})$ obtained by filling the vertices into a d-dimensional grid. Even for planar layouts, each planar layout algorithm embedding the graph in a grid of size $\mathcal{O}(\frac{1}{n}) \times \mathcal{O}(\frac{1}{n})$ (*e.g.* Kant 1996; de Fraysseix *et al.* 1990; Schnyder 1990) yields a resolution of $\mathcal{O}(\frac{1}{n})$.

Theorem 4.1 is readily generalized to graphs with positive edge weights, and likewise the barycentric layout model. It is an interesting open problem, whether the worst case resolution of barycentric layouts can be improved by suitably placed edge weights.

A similar notion of resolution applies to the angles formed by incident edges. Formann *et al.* (1993) define the *angular resolution* of a straight-line representation to be the minimum angle formed by any two incident edges. The visual clutter caused by small angles was already noted in the application of general energy layout methods to time table graphs (Section 3.4). We use the construction of Eades and Garvan (1996) to show that barycentric layouts may also display poor angular resolution.

Corollary 4.5 *The worst case angular resolution of optimal layouts in the barycentric layout model is* $\mathcal{O}(\frac{1}{\alpha^n})$, $\alpha > 1$.

■ **Proof** Consider the family of graphs $G(3), G(4), \ldots$ and associated layouts $x(3), x(4), \ldots$ from the proof of Theorem 4.4. Since the angle formed by edges $\{v_1, v_2\}$ and $\{v_1, v_3\}$ is $\frac{\pi}{4}$ and $x^{(2)}_{v_n}(n) < \frac{1}{3} \cdot x^{(2)}_{v_{n-1}}(n-1)$, the angle formed by edges $\{v_1, v_2\}$ and $\{v_1, v_n\}$ in layout $x(n)$ is at most half the angle formed by edges $\{v_1, v_2\}$ and $\{v_1, v_{n-1}\}$ in layout $x(n-1)$, $n > 3$. □

In Section 5.1.3 it will be shown that this is again poor performance. It is open, however, whether this is true also for planar straight-line representations. Anyhow, up to now no precautions have been taken to control angles resulting from our layout models. The next chapter considers layout methods that do explicitly determine angles prior to assigning positions.

Chapter 5

Angle Flow Layouts

In the previous chapter it was shown that the barycentric layout approach can lead to at least exponentially bad resolution in both distances and angles, even for planar graphs. This was argued to be poor for distances, but it is not known whether it is possible to generate planar layouts with significantly better angular resolution. While there is no control over the angles resulting from barycentric layouts, this chapter focuses on methods to explicitly compute the angles a layout is to display before positioning the vertices.

Intermediate representations or graph transformations are often used to exploit special properties of a graph or representation in algorithmic approaches to graph layout. Clearly, several such steps can be performed one after the other. Di Battista *et al.* (1999) classify graph layout approaches according to the respective sequence of transformations. In particular, they describe the *Topology-Shape-Metrics* approach for polyline representations with axis-parallel edge segments, which is easily generalized to arbitrary polyline representations. Roughly speaking, it consists of the following three steps:

1. *Planarization:* The graph is made planar, *e.g.* by introducing dummy vertices to represent edge crossings of a given embedding. The resulting planar graph might also be augmented to satisfy properties like bi- or triconnectedness required by some layout algorithms (see Kant, 1993, for an extensive discussion). Planarization under several circumstances is discussed by Jünger and Mutzel (1996). In any case, the output is a planar combinatorial embedding, and the topology of the layout is given by this embedding of the planarized graph.

2. *Angle assignment:* Given a planar graph and the ordering of edges around vertices and faces, the angles between incident edges (or edge segments) define the shape of the layout.

3. *Length assignment:* Finally, a feasible length is determined for each edge (or edge segment) in a layout with given topology and shape.

We focus on the second step. Throughout this chapter, we assume that we are given a graph that is planar and embedded, or *plane* for short. We will see that angles in layouts with prespecified topology form a network flow. Unfortunately, not every angle assignment corresponding to a feasible flow in this network yields a planar straight-line representation. Some important special cases, though, are indeed characterized by variants of the network flow model. For instance, flow techniques are used for upward drawings of planar graphs (straight-line representations of graphs in which all directed edges point upwards) in Bertolazzi and Di Battista (1991), and bend-minimum shapes for orthogonal representations of plane graphs with maximum vertex degree at most four are obtained in Tamassia (1987). In an orthogonal representation each edge is represented by an alternating sequence of horizontal and vertical straight line segments, *i.e.* all angles are multiples of $\frac{\pi}{2}$.

The basic network flow model is introduced in Section 5.1, and some of its properties and limitations are discussed. In particular, we prove in Section 5.1.3 that, given a triconnected planar graph that is to be represented on the sphere, angle assignments with asymptotically optimal angular resolution can be obtained in the flow model. Since this model does not necessarily yield realizable shapes, however, this result is only of theoretical interest. The bend-minimum shape model of Tamassia is put in context in Section 5.2. Following the Bayesian framework introduced in Section 2.2, a general tool for the extension of network based orthogonal shape implementations to dynamic graphs is introduced in Section 5.3.

5.1 Angles in Plane Graphs

Shapes of planar straight-line representations of plane graphs are given by angles between each pair of consecutive edges of a face. We index an angle with the pair consisting of its vertex and the face it opens to. In this section, a network flow model for angle assignment is developed, and its limitations with respect to straight-line or great-circle representations are discussed.

5.1.1 Angle Networks

Let $G = (V, E, F)$ be a plane graph. The layout elements in this chapter are angles between pairs of edges or edge segments. For ease of notation, we use variables $x_{(v,f)} \in [0, 2\pi]$ to denote the angle between consecutive edges of a face $f \in F$ sharing a vertex $v \in V$. It will be clear from context, which angle

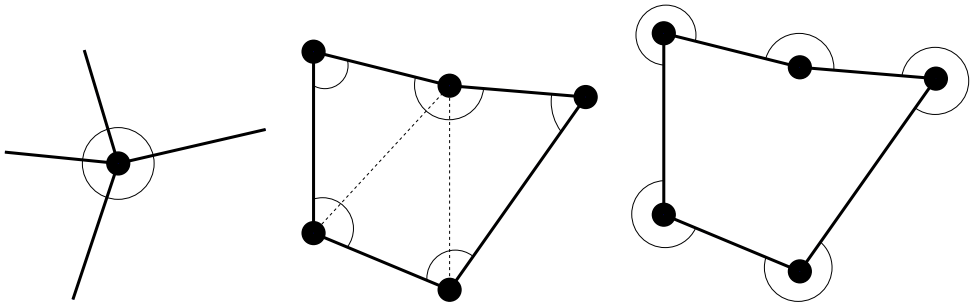


Figure 5.1: Conditions for locally consistent angles illustrated

is referred to in case v is a cutvertex, *i.e.* a vertex whose removal disconnects the graph. Bend angles will be represented differently (see Section 5.2.1). Two obvious conditions are necessary for an angle assignment to describe the shape of a planar straight-line representation (Vijayan 1986; Malitz and Papakostas 1994; see Figure 5.1):

1. *Vertex constraints:* The angles around each vertex must sum up to 2π,

$$\sum_{f \in F \text{ incident to } v} x_{(v,f)} = 2\pi \qquad \text{for all } v \in V$$

2. *Face constraints:* Since each internal face f with $d_G(f)$ edges must form a polygon with $d_G(f)$ vertices, the angles around this face sum up to $(d_G(f)-2)\cdot\pi$,

$$\sum_{v \in V \text{ incident to } f} x_{(v,f)} = (d_G(f)-2)\cdot\pi \qquad \text{for all } f \in F - f_0$$

   Clearly, the angles between consecutive edges of the external face correspond to the external angles of a polygon with $d_G(f_0)$ vertices,

$$\sum_{v \in V \text{ incident to } f_0} x_{(v,f_0)} = d_G(f_0)\cdot 2\pi - (d_G(f_0)-2)\cdot\pi = (d_G(f_0)+2)\cdot\pi$$

The angle assignment of a plane graph is said to be *locally consistent*, if it satisfies all vertex and face constraints. The pairs of vertices and faces forming the indices of variables suggest to think of angles as a commodity that is produced at vertices and consumed in faces. This view leads directly to an elegant formulation of the angle assignment problem in terms of network flow.

A *flow network* is a directed graph $\mathcal{N} = (W, A)$ together with a *demand/supply* function $b : W \to \mathbb{R}$ that satisfies the *total mass balance constraint*, $\sum_{w \in W} b(w) = 0$, *lower capacities* $l : A \to \mathbb{R}$, and *upper capacities* $u : A \to \mathbb{R}$. To distinguish between networks and other graphs, we call the elements of W the *nodes*, and the elements of A the *arcs* of the network. A node u with $b(w) > 0$ has a *supply* of $b(w)$, while a node w with $b(w) < 0$ has a *demand* of $b(w)$.

A vector $x = (x_a)_{a \in A}$ is a *flow*, if it satisfies the *capacity constraints* $l(a) \leq x_a \leq u(a)$ for all $a \in A$, and the *flow conservation constraints*

$$b(w) = \sum_{(w',w) \in A} x_{(w',w)} - \sum_{(w,w') \in A} x_{(w,w')}$$

for all $w \in W$. If $\sum_{(w',w) \in A} x_{(w',w)} = \sum_{(w,w') \in A} x_{(w,w')}$ for all $w \in W$, x is called a *circulation*. In a *positive* flow, $x_a > 0$ for all $a \in A$. For an abundant reference on numerous topics related to flow networks see Ahuja *et al.* (1993).

We can now easily think of angles satisfying the vertex and face constraints as a flow commodity distributed through a fairly simple network.

Definition 5.1 (Angle network) *Let $G = (V, E, F)$ be a plane graph. The* angle network $\mathcal{A}(G) = (W, A; b, l, u)$ *with respect to a designated external face f_0 is defined by*

$$\begin{aligned} W &= V \cup F \\ A &= \{(v, f) \in (V \times F) : v \text{ incident to } f\} \end{aligned}$$

$$\begin{aligned} b(v) &= 2\pi && \text{for all } v \in V \\ b(f) &= -(d_G(f) - 2) \cdot \pi && \text{for all } f \in F - f_0 \\ b(f_0) &= -(d_G(f_0) + 2) \cdot \pi \end{aligned}$$

$$\begin{aligned} l(a) &= 0 && \text{for all } a \in A \\ u(a) &= 2\pi && \text{for all } a \in A \end{aligned}$$

where the elements of A have appropriate multiplicity.

Angle networks thus reflect that each vertex has a supply of 2π radians which is to be distributed among the incident faces. Each face demands the amount of radians appropriate for a polygon with a number of vertices equal to the degree of the face. The demand/supply of an angle network satisfies

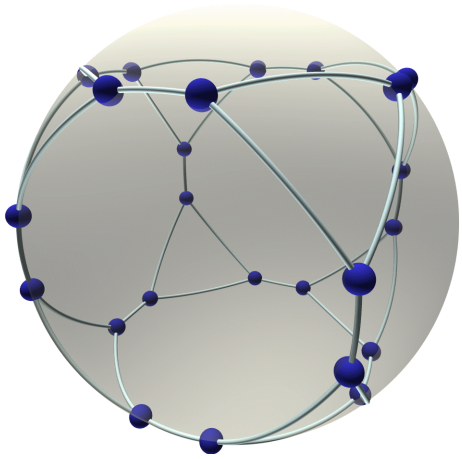


Figure 5.2: Great-circle representation of the graph from Figure 4.1 (optimal layout in a modified barycentric model)

the total mass balance constraint,

$$\begin{aligned}
\sum_{w \in W} b(w) &= \sum_{v \in V} b(v) + \sum_{f \in F} b(f) \\
&= |V| \cdot 2\pi - \left(\sum_{f \in F \setminus f_0} (d_G(f) - 2) \right) \cdot \pi - (d_G(f_0) + 2) \cdot \pi \\
&= |V| \cdot 2\pi - \left(\sum_{f \in F} d_G(f) \right) \cdot \pi + 2(|F| - 1) \cdot \pi - 2 \cdot \pi \\
&= |V| \cdot 2\pi - 2|E| \cdot \pi + |F| \cdot 2\pi - 2 \cdot 2\pi \\
&= 0
\end{aligned}$$

since $\sum_{f \in F} d_G(f) = 2 \cdot |E|$ and $|V| - |E| + |F| = 2$ (Euler's formula).

Every planar polyline representation of a graph in the plane is a distorted representation of the graph in the sense that one face occupies an infinite area. It seems more appropriate to represent planar graphs in a closed surface, particularly a sphere. The analog of a straight line connecting two points in the plane is the shorter segment of a great-circle connecting two non-diametral points on the sphere. The *great-circle* of a pair of non-diametral points on the sphere is the intersection of the sphere with the unique plane defined by these two points and the center of the sphere. Figure 5.2 shows a great-circle representation of the graph from Figure 4.1.

Angles of great-circle representations can be described in the same way as in straight-line representations, provided the face constraints are adjusted. Girard's theorem states that the sum of the angles in a spherical polygon is always greater than or equal to the sum of the angles in a plane polygon with the same number of vertices. Moreover, the difference between these two sums equals the area of the spherical polygon divided by the squared radius of the sphere. This difference is called the *spherical excess*. Since the area of the unit sphere equals 4π, the *spherical face constraints* read

$$\sum_{v \in V \text{ incident to } f} x_{(v,f)} \geq (d_G(f) - 2) \cdot \pi \qquad \text{for all } f \in F$$

and

$$\sum_{f \in F} \sum_{v \in V \text{ incident to } f} \big(x_{(v,f)} - (d_G(f) - 2) \cdot \pi\big) = 4\pi$$

Definition 5.2 (Spherical angle network) *The* spherical angle network $\mathcal{S}(G)$ *of a plane graph* $G = (V, E, F)$ *equals* $\mathcal{A}(G)$*, except that*

$$b(f) \;=\; -\big((d_G(f) - 2) \cdot \pi + \varepsilon_f\big) \quad \textit{for all } f \in F$$

with parameters $\varepsilon_f \geq 0$ *for all* $f \in F$*, where* $\sum_{f \in F} \varepsilon_f = 4\pi$*.*

Again, the total mass balance constraints are easily verified:

$$\begin{aligned}
\sum_{w \in W} b(w) &= \sum_{v \in V} b(v) + \sum_{f \in F} b(f) \\
&= |V| \cdot 2\pi - \left(\sum_{f \in F} d_G(f)\right) \cdot \pi + 2|F| \cdot \pi - \sum_{f \in F} \varepsilon_f \\
&= |V| \cdot 2\pi - 2|E| \cdot \pi + |F| \cdot 2\pi - 2 \cdot 2\pi \\
&= 0
\end{aligned}$$

Flows in $\mathcal{A}(G)$ or $\mathcal{S}(G)$ are called *angle flows* or *spherical angle flows*, respectively. Clearly, (spherical) angle flows are exactly those angle assignments satisfying all vertex and (spherical) face constraints. Network flows can therefore be used to study feasibility problems related to angle assignments. While the remainder of this section is devoted to this kind of problems, we will see in Section 5.2 that, by imposing a cost function, network flow techniques can also be used to additionally guarantee certain properties of angle assignments.

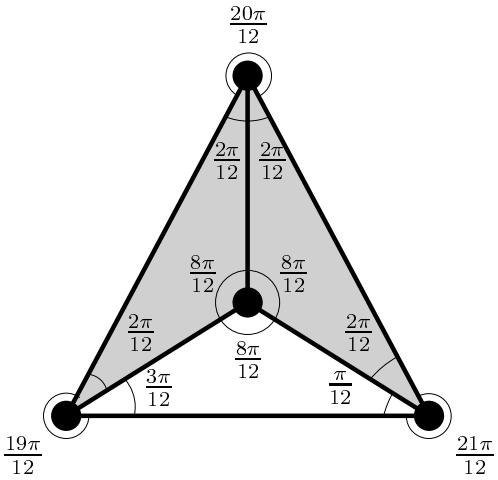


Figure 5.3: A plane graph that has no straight-line representation preserving the indicated angle assignment (note that the shaded faces form isosceles with a common side)

5.1.2 Planar Realizability

The question immediately arising from the above observations is, whether the shape of planar straight-line or great-circle representations of plane graphs can be determined by using standard network flow techniques?

Unfortunately, the answer is no. Consider the angle assignment for a plane K_4, the complete graph on four vertices, in Figure 5.3. Although all vertex and face constraints are satisfied, there is no straight-line representation preserving these angles. The counterexample is easily generalized to larger graphs and great-circle representations.

An angle assignment for an embedded graph is said to be *(planar) realizable*, if there is an angle preserving (planar) straight-line or great-circle representation. Let us summarize the above definitions and observations.

Lemma 5.3 (Angle Flow Lemma) *An angle assignment x for a plane graph G is locally consistent, if and only if x is a flow in $\mathcal{A}(G)$ or $\mathcal{S}(G)$, respectively. Infinitely many plane graphs have locally consistent angle assignments that are not realizable.*

Garg (1995) proves that it is $\mathcal{NP}$-complete to decide whether a locally consistent angle assignment is planar realizable, even if all angles are multiples of $\frac{2\pi}{k}$ for some integer k. Planar realizability can be tested efficiently for special graph classes, *e.g.* series-parallel graphs (Garg, 1995) or trees (trivial). Di Battista and Vismara (1996) give a compact characterization of planar realizability for the following important special case. In particular, it shows

the necessity of a non-linear condition that is not implied. This fact will be the snag of our result on spherical angle flows in the next subsection.

Theorem 5.4 (Di Battista and Vismara 1996) *Let $G = (V, E, F)$ be a triconnected plane graph with designated external face f_0 and triangular internal faces, and let x be a flow in $\mathcal{A}(G)$. The corresponding angle assignment gives rise to a planar straight-line representation that represents the external face by a convex polygon, if and only if*

(i) For all vertices v incident to f_0

$$\sum_{f \in F - f_0 \text{ incident to } v} x_{(v,f)} \leq \pi$$

(ii) Ceva constraints: *For all vertices v not incident to f_0*

$$\prod_{f \in F \text{ incident to } v} \frac{\sin x_{(v_f^{\leftarrow},f)}}{\sin x_{(v_f^{\rightarrow},f)}} = 1$$

where $v_f^{\leftarrow}$ and $v_f^{\rightarrow}$ are the other two vertices incident to the same face f in clockwise order (see Figure 5.4).

This characterization is minimal in the sense that there is an infinite family of admissible graphs with angle assignments that violate only condition (ii) for a single vertex, but are not planar realizable.

Necessity of the first condition is trivial. We suggest calling the second one Ceva constraint, since its necessity can be derived from Ceva's theorem[1] (Sigl, 1969, p. 121).

5.1.3 Angular Resolution

Angular resolution was defined in Section 4.1.2 to quantify a weakness of the barycentric layout model. One might conjecture that, analogous to the case of spatial resolution, tailor-made methods can substantially improve angular resolution. Methods based on realizable angle graphs appear to be natural candidates. However, maximizing angular resolution in planar straight-line representations is $\mathcal{NP}$-hard, even for triconnected planar graphs

[1]Ceva's theorem is as follows: If points P, Q, and R are on the sides BC, CA, and AB of a triangle ABC, respectively, then the lines AP, BQ, and CR meet in one point, if and only if $\frac{BP}{PC} \cdot \frac{CQ}{QA} \cdot \frac{AR}{RB} = 1$.

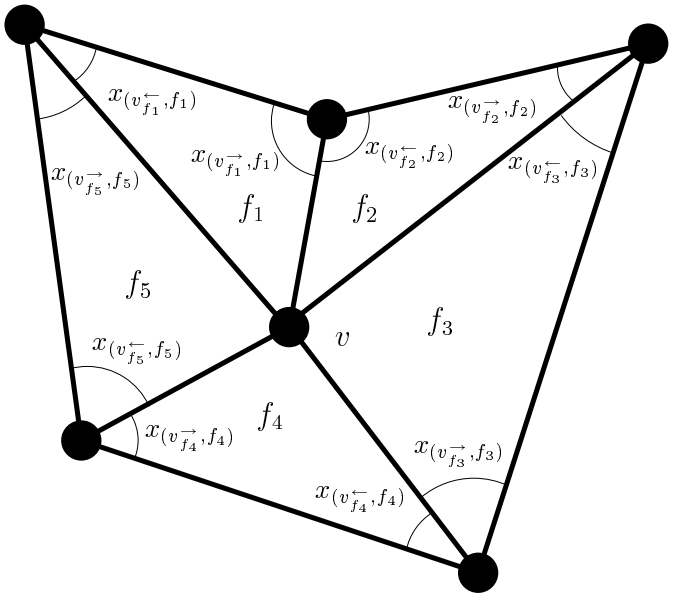


Figure 5.4: The Ceva constraint: $\prod_{f \in F \text{ incident to } v} \frac{\sin x_{(v^{\leftarrow}_f,f)}}{\sin x_{(v^{\rightarrow}_f,f)}} = 1$

(Garg, 1996). Moreover, even the problem of proving tight bounds on the angular resolution is largely open.

We give a brief overview of known results for planar straight-line representations, and then extend a theorem of Malitz and Papakostas (1994) on a lower bound for the angular resolution of triangulated plane graphs to spherical layouts. While the angle assignments thus provided display very appealing additional properties, we have to admit that there is no practical use for them but to stress the gap between local consistency and realizability of angle assignments.

A trivial upper bound on the angular resolution is $\frac{2\pi}{\Delta(G)}$. Formann *et al.* (1993) show that every planar graph has a straight-line representation achieving this bound (up to a constant factor), but the resulting representation is not necessarily planar.[2] Lower bounds on the angular resolution of straight-line representations preserving planarity are given by Malitz and Papakostas (1994) and Garg and Tamassia (1994). While Malitz and Papakostas (1994) prove an existential lower bound of $\Omega\left(\frac{1}{\alpha^{\Delta(G)}}\right)$, for some $\alpha > 1$, using the fact that every triconnected planar graph has a coin graph representation (see Figure 2.4(b)), Garg and Tamassia (1994) show that this bound is not tight.

[2]In fact, they also show that every graph has a (in general non-planar) straight-line representation with angular resolution $\mathcal{O}\left(\frac{1}{\Delta(G)^2}\right)$.

They also give the best known upper bound of $\mathcal{O}\left(\sqrt{\frac{\log \Delta(G)}{\Delta(G)^3}}\right)$ for planar straight-line representations of planar graphs.

Independent of the maximum vertex degree, every algorithm placing a plane graph in a grid of size $\mathcal{O}(n) \times \mathcal{O}(n)$ yields a lower bound of $\Omega\left(\frac{1}{n}\right)$ (*e.g.* Kant 1996; de Fraysseix *et al.* 1990; Schnyder 1990).

While it is an open problem whether a plane graph has a planar straight-line representation with angular resolution polynomial in $\frac{1}{\Delta(G)}$, note that, in contrast, it is possible to achieve asymptotically optimal angular resolution $\Omega\left(\frac{1}{\Delta(G)}\right)$ in planar polyline representations (Kant, 1996), *i.e.* by allowing edges to bend.

A plane graph is *triangulated*, if every face is incident to exactly three vertices. Malitz and Papakostas (1994, p. 173) speculate "that $\Omega\left(\frac{1}{\Delta(G)}\right)$ ... is perhaps the true lower bound on angular resolution for triangulated planar graphs" by proving the theorem below. Since every simple planar graph can be triangulated such that the maximum vertex degree increases only by a constant factor (see, *e.g.*, Kant, 1993, Chapter 6), this would also imply the asymptotically optimal lower bound for all simple plane graphs.

Theorem 5.5 (Malitz and Papakostas 1994) *Every triangulated plane graph has an angle flow with minimum angle* $\Omega\left(\frac{1}{\Delta(G)}\right)$.

Recall that the angles of a planar straight-line representation form a flow in the corresponding angle network, while the converse is not necessarily true – and even $\mathcal{NP}$-complete to decide (Garg, 1995). Also recall that, by now, an upper bound of $\mathcal{O}\left(\sqrt{\frac{\log \Delta(G)}{\Delta(G)^3}}\right)$ for the angular resolution of planar straight-line representations has been proven by Garg and Tamassia (1994).

In the remainder of this section we show that, along the lines of Malitz and Papakostas (1994), one might just as well speculate about the existence of very special planar great-circle representations of triconnected planar graphs. In particular, they would match the trivial upper bound on the angular resolution up to a constant factor. Note that every triangulated plane graph is triconnected (Goldschmidt and Takvorian, 1994).

We first define the notion of an incidence graph to capture the relation between a plane multigraph and its *dual*, *i.e.* the multigraph that has the faces of G as vertices, and, for every edge in the original graph, an edge connecting the vertices corresponding to the faces the edge is incident to. Then we extend the proof of Malitz and Papakostas (1994) to spherical angle flows of incidence graphs of triconnected planar graphs.

The associated *incidence graph* $\mathcal{I}(G) = (V_\mathcal{I}, E_\mathcal{I}, F_\mathcal{I})$ of a plane graph G is the plane graph which has a vertex for each vertex, face, and edge of G. Two

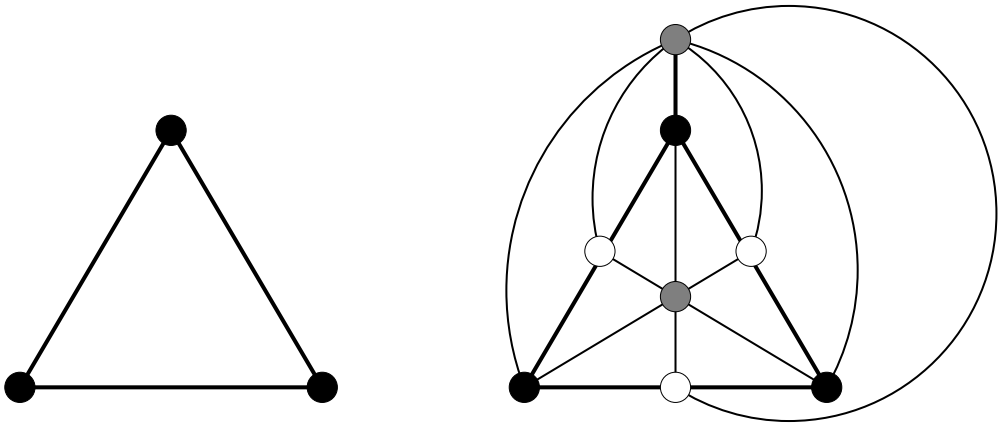


Figure 5.5: A plane graph and its associated incidence graph

vertices of $\mathcal{I}(G)$ are adjacent, if the corresponding elements of G are incident, and the embedding of $\mathcal{I}(G)$ refines the embedding of G (cf. Figure 5.5). Note that the incidence graphs of G and its dual are isomorphic.

Incidence graphs represent embeddings of planar graphs. Moreover, a geometric embedding of $\mathcal{I}(G)$ gives rise to a simultaneous geometric embedding of G and its dual. Obviously, the incidence graph of a simple triconnected planar graph is simple, and every incidence graph is triangulated.

We conclude the introduction of incidence graphs with a lemma that will be useful later on. Suppose we are given a plane graph G and a simple cycle ζ in $\mathcal{I}(G)$. When ζ is traversed in clockwise direction, its right hand side is called the *interior*, while its left hand side is called the *exterior* of ζ.

Lemma 5.6 *Let G be a triconnected planar graph, and let ζ be a simple cycle of length r in $\mathcal{I}(G)$. If $r < 6$, then the interior or the exterior of ζ contains at most one vertex of $\mathcal{I}(G)$. This vertex then corresponds to an edge in G.*

■ **Proof** The removal of ζ disconnects the interior from the exterior. Clearly, $r > 2$. If $r = 3$, then the interior or the exterior is empty. Assume therefore that $4 \leq r \leq 5$. Since no two vertices of the same type can be adjacent, ζ contains at most two vertices corresponding to vertices, edges, and faces in G, respectively. Since a planar graph is triconnected if and only if its dual is, the interior or the exterior of ζ does not contain a vertex or face of G. Finally verify that neither interior nor exterior can contain two vertices corresponding to edges without containing a vertex corresponding to a vertex or face of G. □

Our aim now is to show the existence of locally consistent spherical angle assignments for incidence graphs $\mathcal{I}(G)$ of triconnected planar graphs G, such that every angle at some vertex v is in $\Theta\left(\frac{1}{d_G(v)}\right)$. Analogous to the construction in the original proof of Theorem 5.5, we investigate a flow problem in the spherical angle network $\mathcal{S}(\mathcal{I}(G))$ of $\mathcal{I}(G)$.

Since we want to guarantee a minimum flow value for each arc in the spherical angle network $\mathcal{S}(\mathcal{I}(G)) = (W, A; b, l, u)$, parameters $l(v, f) = \alpha_v \in \left(0, \frac{2\pi}{d_{\mathcal{I}(G)}(v)}\right]$ are introduced for all $(v, f) \in A$. They express that every angle at v shall have minimum value $\alpha_v > 0$. Recall that $\frac{2\pi}{d_{\mathcal{I}(G)}(v)}$ is an upper bound on the minimum angle at vertex $v \in V_{\mathcal{I}}$. With $u(v, f) = \frac{\pi}{2}$ for all $(v, f) \in A$ the capacities are well-defined because every vertex in the incidence graph of a triconnected planar graph has degree at least 4, and hence $0 < \alpha_v \leq \frac{\pi}{2}$. See Figure 5.6.

To prove the existence of a flow in $\mathcal{S}(\mathcal{I}(G))$ for certain values of α_v, $v \in V_{\mathcal{I}}$, and ε_f, $f \in F_{\mathcal{I}}$ we need some more notation. For a subset of nodes $S \subseteq W$, let $S_V = S \cap V_{\mathcal{I}}$ and $S_F = S \cap F_{\mathcal{I}}$ be the subsets of nodes corresponding to vertices and faces in $\mathcal{I}$, respectively. Furthermore, let $F_i(S_V) \subseteq F_{\mathcal{I}}$ be the set of faces incident to exactly i, $0 \leq i \leq 3$, vertices in S_V. Any non-empty, proper subset $S \subset W$ of nodes in a network is called a *cut*. The *margin* $m(S)$ of a cut $S \subset W$ is the capacity available for flow leaving S,

$$m(S) = \sum_{a \in A \cap (S \times (W \setminus S))} u(a) - \sum_{a \in A \cap ((W \setminus S) \times S)} l(a) - \sum_{w \in S} b(w)$$

A cut S is called *tight*, if $S_F = F_2(S_V) \cup F_3(S_V)$ and the induced subgraph $\mathcal{I}(G)[V_{\mathcal{I}} \setminus S_V]$ contains no isolated vertex, *i.e.* the boundary of the subgraph of $\mathcal{I}(G)$ induced by S_V consists of simple cycles that do not isolate single vertices.

Lemma 5.7 *Let $G = (V, E, F)$ be a triconnected plane graph and $\varepsilon_f \leq \alpha_v \leq \frac{2\pi}{4 + d_{\mathcal{I}(G)}(v)}$ for all $f \in F$ and $v \in V$. For every cut $S \subset W$ in $\mathcal{S}(\mathcal{I}(G))$, there is a tight cut with smaller or equal margin.*

■ **Proof** If a face f is removed from a cut S, the margin of the cut changes by less than $3 \cdot \frac{\pi}{4} - (\pi + \varepsilon_f) < 0$ if $f \in F_0(S_V)$, and by less than $\frac{\pi}{2} + 2 \cdot \frac{\pi}{4} - (\pi + \varepsilon_f) \leq 0$ if $f \in F_1(S_V)$. Likewise it does not increase if a face f incident to at least two vertices in S_V is added to S.

Assume therefore that $S_F = F_2(S_V) \cup F_3(S_V)$ and that there is an isolated vertex v in $\mathcal{I}(G)[V_{\mathcal{I}} \setminus S_V]$. Since all neighbors of v in $\mathcal{I}(G)$ must be in S_V, every face incident to v is in $F_2(S_V)$. It follows from $\sum_{f \text{ incident to } v} \alpha_v < 2\pi$ that $m(S \cup \{v\}) < m(S)$. □

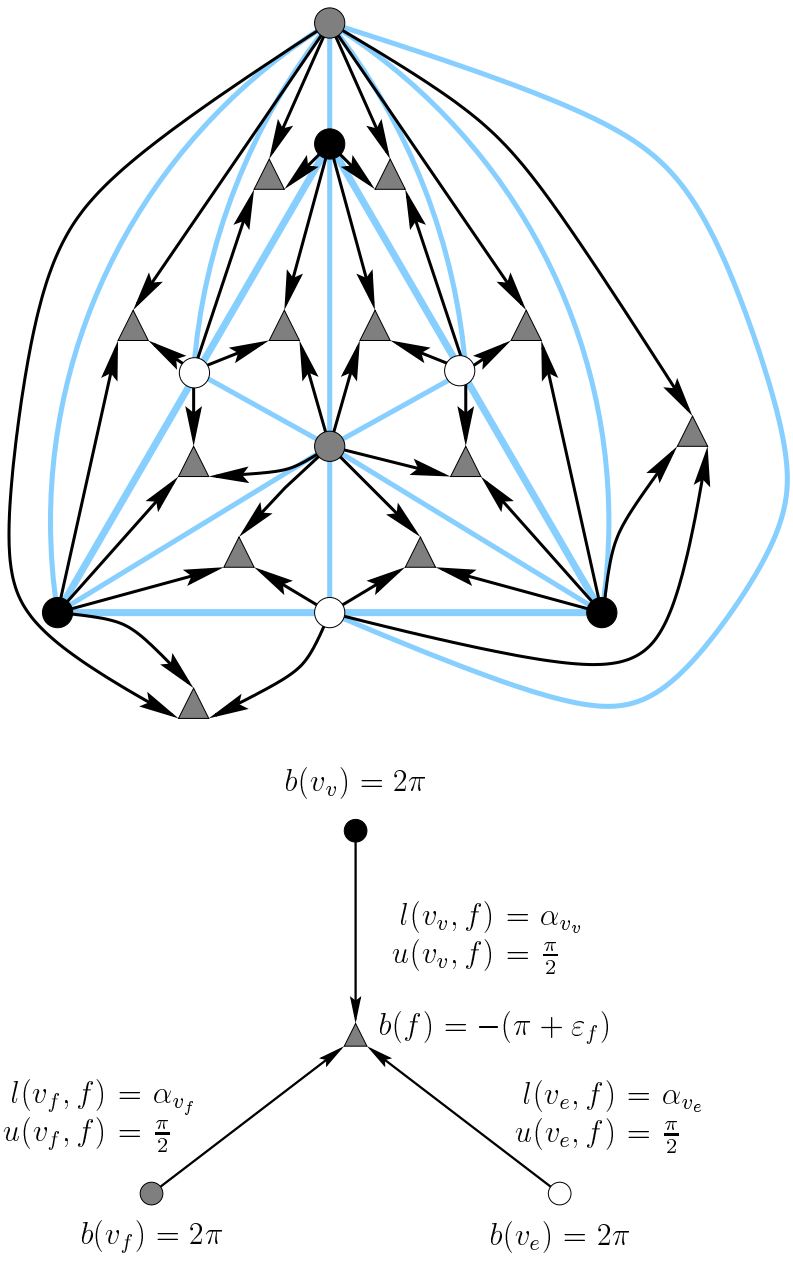


Figure 5.6: Spherical angle network for the incidence graph in Figure 5.5. The bottom figure shows the network parameters for a face f with incident vertices v_v, v_f, and v_e of the incidence graph (corresponding to a vertex, face, and edge, respectively, in the original graph)

For a tight cut $S \subseteq W$, we define its *closure* to be the subgraph of $\mathcal{I}(G)$ $\mathcal{I}_S$ of S consisting of all vertices and edges incident to at least one face in $F_1(S_V) \cup F_2(S_V) \cup F_3(S_V)$.

Lemma 5.8 *Let G be a plane graph and $S \subseteq W$ be a tight cut in $\mathcal{S}(\mathcal{I}(G))$. If the closure $\mathcal{I}_S$ of S consist of biconnected components $\mathcal{I}_S^{(1)}, \ldots, \mathcal{I}_S^{(k)}$, then*

$$m(S) = m(S^{(1)}) + \cdots + m(S^{(k)})$$

where $S^{(1)}, \ldots, S^{(k)}$ is the corresponding partition of S.

■ **Proof** Since S is tight, no cutvertex of $\mathcal{I}_S$ is in S. Hence, the obvious association of subsets $S^{(i)} \subseteq S$ to components $\mathcal{I}^{(i)}$, $i = 1, \ldots, k$, gives indeed a partition, and no node in some $S^{(i)}$ corresponding to a vertex in $\mathcal{I}(G)$ is incident to a node in some $S^{(j)}$, $j \neq i$, corresponding to a face. □

Theorem 5.9 *Let G be a triconnected planar graph on three or more vertices. There is a flow in $\mathcal{S}(\mathcal{I}(G))$, if $\alpha_v = \frac{\pi}{6 \cdot \left(d_{\mathcal{I}(G)}(v) - 3\right)}$ for all $v \in V_{\mathcal{I}}$ and $\varepsilon_f = \frac{4\pi}{|F_{\mathcal{I}}|}$ for all $f \in F_{\mathcal{I}}$.*

■ **Proof** The planar embedding of a triconnected graph G is unique, so that $\mathcal{I}(G) = (V_{\mathcal{I}}, E_{\mathcal{I}}, F_{\mathcal{I}})$ and $\mathcal{S}(\mathcal{I}(G)) = (W, A; b, l, u)$ are well-defined. By the Ford/Fulkerson-Theorem and Lemmas 5.7 and 5.8, there is a flow in $\mathcal{S}(\mathcal{I}(G))$, if the margin of every tight cut with biconnected closure is non-negative. Let $S \subset W$ be any tight cut with biconnected closure $\mathcal{I}_S$. Then,

$$m(S) = \sum_{\substack{(v,f) \in A: \\ v \in S_V, f \in F_1(S_V)}} \frac{\pi}{2} - \sum_{\substack{(v,f) \in A: \\ v \in V_{\mathcal{I}} \setminus S_V, f \in F_2(S_V)}} \alpha_v - \sum_{v \in S_V} 2\pi + \sum_{f \in S_F} (\pi + \varepsilon_f)$$

To bound $m(S)$ from below, we are interested in bounds on the number of faces in $F_1(S_V)$ and $F_2(S_V)$. Let $\partial\mathcal{I}_S$ be the subgraph of $\mathcal{I}_S$ induced by vertices not in S. We call $\partial\mathcal{I}_S$ the *boundary* of $\mathcal{I}_S$. The boundary of a biconnected closure $\mathcal{I}_S$ is a planar graph with all vertices incident to a common face and without chords. Since S is tight, each connected component of $\partial\mathcal{I}_S$ has at least two vertices. The number of edges encountered when walking around each component of $\partial\mathcal{I}_S$ is a lower bound on the number of faces in $F_1(S_V)$. An upper bound for the number of faces in $F_2(S_V)$ can be determined from the degrees of the boundary vertices. These arguments are still valid when closure $\mathcal{I}_S$ is modified in the following way. Each component of $\partial\mathcal{I}_S$ is transformed into a simple cycle by duplicating edges that are encountered twice when walking around its component, and then splitting each vertex according to the number of simple boundary cycles it is incident

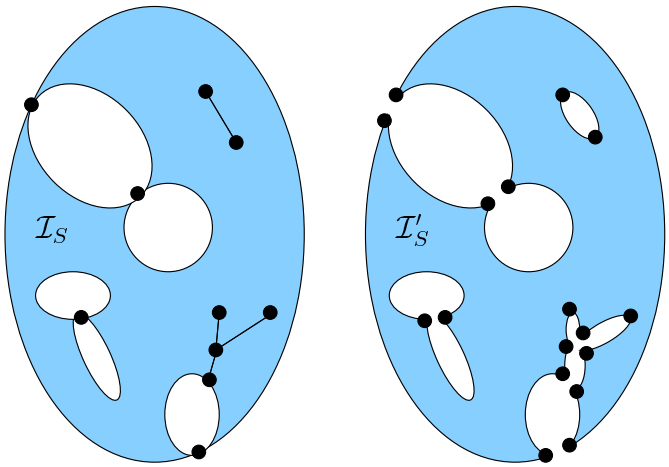


Figure 5.7: Transforming each component of $\partial\mathcal{I}_S$ into a simple cycle by duplicating edges and splitting vertices

to. We denote the modified graph with $\mathcal{I}'_S$ and its boundary with $\partial\mathcal{I}'_S$ (see Figure 5.7).

Let $\partial\mathcal{I}'_S$ consist of $k \geq 1$ simple cycles, and let $r \geq 2$ be the number of boundary edges of $\mathcal{I}'_S$. Observe that $\mathcal{I}'_S$ can be triangulated by adding k new vertices, one into each component of $\partial\mathcal{I}'_S$, and connecting them to each vertex of their respective cycle. The resulting triangular graph has $|S_V|+r+k$ vertices and therefore $2\left(|S_V|+r+k\right)-4$ faces. Moreover, $F_1(S_V)$ contains at least r faces, while $F_2(S_V)$ contains no more than $\sum_{v\in V_{\partial\mathcal{I}'_S}} \left(d_{\mathcal{I}(G)}(v)-3\right)$ faces, where $V_{\partial\mathcal{I}'_S}$ is the set of boundary vertices. It follows that

$$\begin{aligned} m(S) &\geq r\cdot\frac{\pi}{2} - \sum_{v\in V_{\partial\mathcal{I}'_S}} \left(d_{\mathcal{I}(G)}(v)-3\right)\alpha_v - |S_V|\cdot 2\pi \\ &\quad + \left[\,2\left(|S_V|+r+k\right)-4-2r\,\right]\pi + \sum_{f\in S_F}\varepsilon_f \\ &= r\cdot\frac{\pi}{2} - r\cdot\frac{\pi}{6} + 2(k-2)\pi + \sum_{f\in S_F}\varepsilon_f \\ &= \left(\frac{r}{3}+2k-4\right)\cdot\pi + \sum_{f\in S_F}\varepsilon_f \end{aligned}$$

which is positive for $k \geq 2$. Therefore we henceforth assume that $k = 1$, *i.e.* $\mathcal{I}'_S$ has exactly one boundary cycle. Since we are done, if $r \geq 6$, we also assume that $r < 6$.

It follows from Lemma 5.6 that either S_V or $V_{\mathcal{I}} \setminus V_{\mathcal{I}'_S}$ contains at most one

vertex, where $V_{\mathcal{I}'_S}$ denotes the set of vertices of the modified closure. If S_V contains at most one vertex, say $v \in V_{\mathcal{I}}$, then $S = \{v\}$, since S is tight. In this case, $m(S) = d_{\mathcal{I}(G)}(v) \cdot \frac{\pi}{2} - 2\pi \geq 0$, since G is triconnected and therefore $d_{\mathcal{I}(G)}(v) \geq 4$.

Finally assume that $V_{\mathcal{I}} \setminus V_{\mathcal{I}'_S}$ contains at most one vertex. Since G has at least three vertices, $\mathcal{I}(G)$ has at least 8 vertices and 12 faces (see Figure 5.5). On the other hand, there are at most $2r$ faces in $F_{\mathcal{I}} \setminus S_F = F_1(S_V) \cup F_0(S_V)$, which gives

$$\begin{aligned} m(S) &\geq \left(\frac{r}{3} - 2\right) \cdot \pi + \sum_{f \in S_F} \varepsilon_f \\ &\geq \left(\frac{r}{3} - 2\right) \cdot \pi + (|F_{\mathcal{I}}| - 2r)\frac{4\pi}{|F_{\mathcal{I}}|} \\ &= \left[2 - \left(\frac{8}{|F_{\mathcal{I}}|} - \frac{1}{3}\right) \cdot r\right] \cdot \pi \\ &\geq \left(2 - \frac{r}{3}\right) \cdot \pi \quad > \quad 0 \end{aligned}$$

□

Corollary 5.10 *The incidence graph $\mathcal{I}(G)$ of any triconnected planar graph G has a spherical angle assignment such that*

- *angles at $v \in V_{\mathcal{I}}$ are in $\Omega\left(\frac{1}{d_{\mathcal{I}(G)}(v)}\right)$ and less than or equal to $\frac{\pi}{2}$*
- *angles at vertices corresponding to edges in G equal $\frac{\pi}{2}$*
- *the angles of each face sum up to $\pi + \frac{4\pi}{|F_{\mathcal{I}}|}$*

If such an angle assignment admits a great-circle representation on the sphere, it is the refinement of simultaneous planar great-circle representations of G and its dual with the following properties:

- the angular resolution of both G and its dual is asymptotically optimal
- primal and corresponding dual edges cross at right angles
- primal and dual faces are convex
- primal and dual faces have area proportional to their degree

That's clearly too much to hope for. Indeed it is easy to see that in general an angle assignment resulting from a flow in $\mathcal{S}(\mathcal{I}(G))$ is not realizable.

Note however that Brightwell and Scheinerman (1993) prove the existence of simultaneous planar straight-line and planar great-circle representations of triconnected planar graphs and their duals such that primal and dual edges cross at right angles. And finally, every planar straight-line representation with convex faces (*e.g.* Tutte, 1963; Kant, 1996) yields a planar great-circle representation with convex faces by gnomonic projection (with straightforward modification to ensure convexity of the originally unbounded face).

5.2 Orthogonal Representation

Up to now angle networks appear to be useless for layout computation, because they do not sufficiently characterize planar realizable angle assignments of plane graphs. Even in the important special case of triconnected plane graphs with triangular internal faces additional non-linear constraints were necessary (Theorem 5.4).

The situation changes for the better if all angles are required to be non-zero multiples of $\frac{\pi}{2}$. At first sight, this may appear a rather arbitrary restriction, but representing edges by alternating sequences of horizontal and vertical straight line segments as a way of reducing perceptual complexity in network visualizations has a tradition in many technical applications.[3] Such polyline representations, in which every angle between two incident edge segments is a multiple of $\frac{\pi}{2}$, are called *orthogonal representations*. They appear to be particularly common for Entity-Relationship diagrams as in Figure 5.8. Since it is possible to characterize realizable angle assignments for orthogonal representations by an augmented angle network, the remainder of this chapter is devoted to orthogonal layout of both static and dynamic graphs.

A generic strategy for generating orthogonal representations is the Topology-Shape-Metrics approach mentioned in the introduction to this chapter. In this section, we assume that the topology step has already been performed, *i.e.* we are given a plane graph. It is shown that shapes of orthogonal representations can be determined from flow in an augmented angle network.

In Section 5.2.1 we first describe a data structure that can be used to store orthogonal shapes, and then give the details for an implementation of the shape step minimizing the number of bends. For completeness, we sketch in Section 5.2.2 an implementation of the subsequent metrics step assigning lengths to the edge segments of an orthogonal shape.

[3] Recall our conjecture about the originators of Figure 3.2(a).

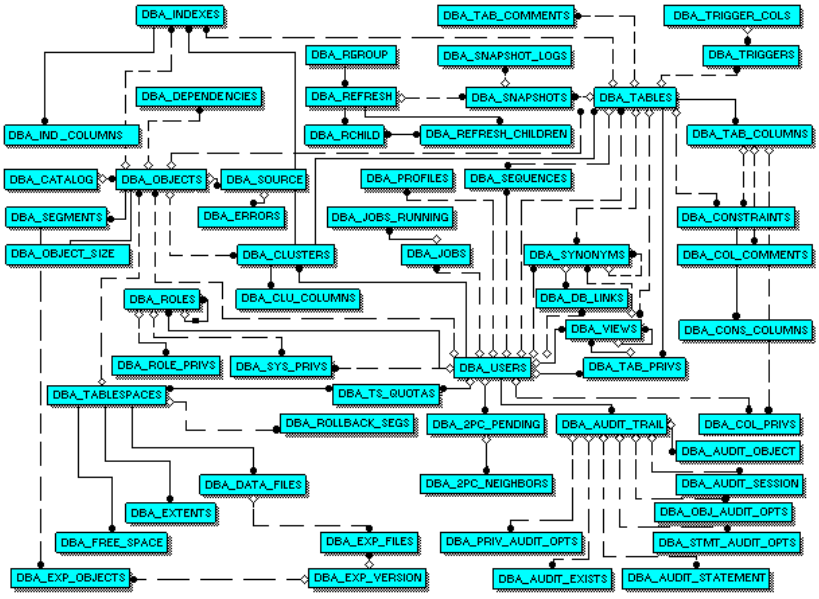


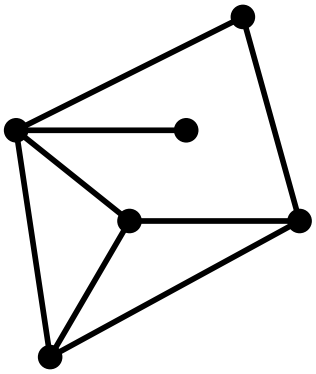
Figure 5.8: Orthogonal representation of an Entity-Relationship diagram as found in the *Web* (RevealNet, Inc.; `http://www.revealnet.com/wpdbaerd.htm`)

5.2.1 Bend-Minimum Shape

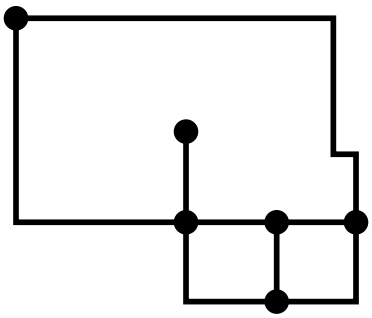
Orthogonal layout with vertices represented by points corresponds to an embedding in a grid, where vertices are placed at grid points and edges are represented by sequences of grid lines. If these edge routes are required to be non-overlapping, the class of admissible graphs is obviously restricted to those with maximum vertex degree at most four. A graph with maximum vertex degree at most k is called a *k-graph*. Since several generalizations of the basic grid embedding approach to graphs of higher degree have been devised (Tamassia *et al.*, 1988; Fößmeier and Kaufmann, 1996; Klau and Mutzel, 1998), we restrict our exposition to the 4-graph case and indicate possible extensions only when appropriate.

Figure 5.9 depicts the three steps of the Topology-Shape-Metrics approach as applied to orthogonal representation. We here describe the bend minimizing implementation of the shape step for plane 4-graphs given by Tamassia (1987). It is extended to dynamic graphs in Section 5.3. Tamassia uses a data structure $\mathcal{H}(G)$ to represent the *orthogonal shape* of a plane 4-graph $G = (V, E, F)$ with designated external face f_0. It consists of circular lists

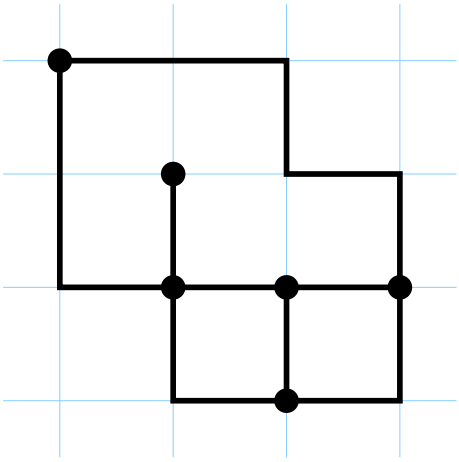
$$H_f = [\,(e_0, s_0, a_0), \ldots, (e_{d_G(f)-1}, s_{d_G(f)-1}, a_{d_G(f)-1})\,]$$



(a) plane 4-graph ("topology")

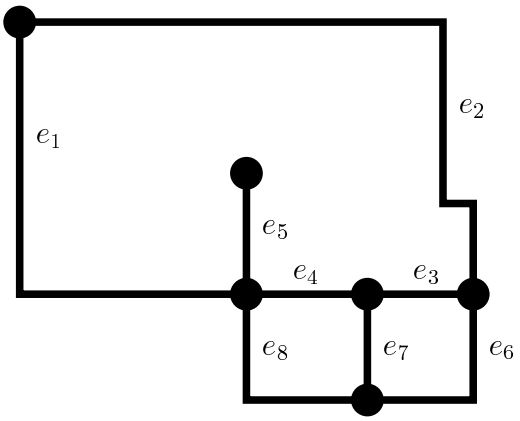


(b) orthogonalization ("shape")



(c) grid embedding ("metrics")

Figure 5.9: The three steps of the Topology-Shape-Metrics approach for orthogonal representations



$[\,(e_1, \mathtt{r}, \frac{\pi}{2}), (e_2, \mathtt{rlr}, \frac{\pi}{2}), (e_3, \epsilon, \pi), (e_4, \epsilon, \frac{\pi}{2}), (e_5, \epsilon, 2\pi), (e_5, \epsilon, \frac{\pi}{2})\,]$
$[\,(e_6, \mathtt{r}, \frac{\pi}{2}), (e_7, \epsilon, \frac{\pi}{2}), (e_3, \epsilon, \frac{\pi}{2})\,]$
$[\,(e_7, \epsilon, \frac{\pi}{2}), (e_8, \mathtt{r}, \frac{\pi}{2}), (e_4, \epsilon, \frac{\pi}{2})\,]$
$[\,(e_1, \mathtt{l}, \frac{\pi}{2}), (e_8, \mathtt{l}, \pi), (e_6, \mathtt{l}, \pi), (e_2, \mathtt{lrl}, \frac{3\pi}{2})\,]$

Figure 5.10: Orthogonal shape from Figure 5.9 with corresponding face lists

for each face $f \in F$. Each triple (e_i, s_i, a_i) of face list H_f consists of an edge $e_i \in E$, a string $s_i \in \{\mathtt{l}, \mathtt{r}\}^*$, and an angle $a_i \in \{\frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi\}$. For every internal face f, the sequence $e_0, \ldots, e_{d_G(f)-1}$ corresponds to a clockwise traversal of the edges incident to f. The external face f_0 is traversed counterclockwise. Recall that each edge is encountered twice, possibly in the same list. In s_i, $\mathtt{l}$'s and $\mathtt{r}$'s represent bends of e_i with angles of $\frac{3\pi}{2}$ (left turn) and $\frac{\pi}{2}$ (right turn) radians towards f, respectively. Finally, a_i is the angle formed by e_i and $e_{i+1 \bmod d_G(f)}$ towards f. Speaking of orthogonal shape, we always assume consistent face lists which do give the angles of an orthogonal representation. Characterization of consistent face lists is straightforward (Tamassia, 1987). An example is given in Figure 5.10.

The angle network defined in Section 5.1 cannot be used to describe orthogonal representations, because even though every bend can be considered a vertex of degree two, the number of bends on an edge is not known beforehand. Consequently, the face constraints are no longer suitable.

A bend of $\frac{\pi}{2}$ radians, corresponding to a right turn in a clockwise traversal of the face, increases the sum of angles required in the facial polygon by π (due to the additional vertex), while it increases the actual sum only by $\frac{\pi}{2}$. It therefore changes the balance of the face node by $-\frac{\pi}{2}$. Likewise, a bend of $\frac{3\pi}{2}$ radians changes the balance by $+\frac{\pi}{2}$. Since each bend of $\frac{\pi}{2}$ radians with respect to one face is a bend of $\frac{3\pi}{2}$ radians with respect to the face on the other side of the edge, a bend causes an amount of $\frac{\pi}{2}$ radians to be transferred

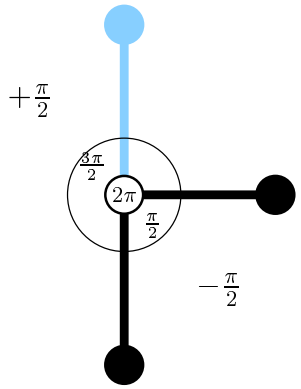


Figure 5.11: A bend in an orthogonal representation effectively shifts an amount of $\frac{\pi}{2}$ from the sum of angles of one face to that of an adjacent face

from one face to an adjacent one without changing the total balance. See Figure 5.11. Bends can hence conveniently be incorporated into an angle network by adding new arcs between adjacent faces. Moreover, using the common divisor $\frac{\pi}{2}$, all parameters can be made integer, so that there always is an integer solution to the flow problem, if any.

Definition 5.11 (Bend and angle network) *Let $G = (V, E, F)$ be a plane 4-graph with designated external face f_0. The* bend and angle network *$\mathcal{B}(G) = (W, A \cup B; b, l, u, c)$ of G is defined by*

$$\begin{aligned}
W &= V \cup F \\
A &= \{(v,f) \in V \times F : v \text{ incident to } f\} \\
B &= \{(f,g) \in F \times F : f \text{ adjacent to } g\}
\end{aligned}$$

$$\begin{aligned}
b(v) &= 4 && \text{for all } v \in V \\
b(f) &= -2 \cdot (d_G(f) - 2) && \text{for all } f \in F - f_0 \\
b(f_0) &= -2 \cdot (d_G(f_0) + 2)
\end{aligned}$$

$$\begin{aligned}
l(v,f) &= 1 && \text{for all } (v,f) \in A \\
u(v,f) &= 4 && \text{for all } (v,f) \in A \\
c(v,f) &= 0 && \text{for all } (v,f) \in A
\end{aligned}$$

$$\begin{aligned}
l(f,g) &= 0 && \text{for all } (f,g) \in B \\
u(f,g) &= \infty && \text{for all } (f,g) \in B \\
c(f,g) &= \gamma && \text{for all } (f,g) \in B
\end{aligned}$$

for some $\gamma \geq 0$. Elements of A and B have the appropriate multiplicity.

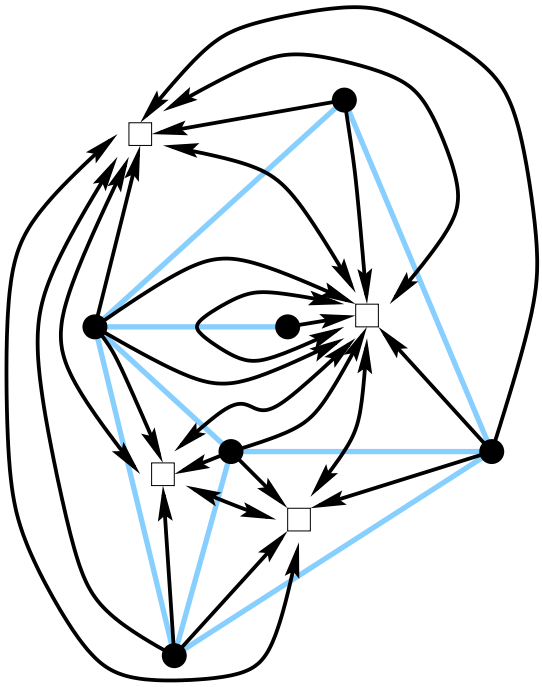


Figure 5.12: Bend and angle network for the plane graph of Figure 5.9

A feasible flow in $\mathcal{B}(G)$ is called a *bend and angle flow*. Each unit of flow represents an angle of $\frac{\pi}{2}$ which is distributed from a vertex to a face and, possibly, to other faces until it is consumed in a face (cf. Figure 5.12). Note that each time a flow unit passes an arc connecting two face nodes, it produces a bend. It has been shown by experiment that diagrams are more ergonomic if the number of bends is small (Purchase *et al.*, 1996). The use of arcs between adjacent faces is therefore discouraged by introducing the *cost function* $c : A \cup B \to \mathbb{R}$. To minimize the number of bends, it is then sufficient to determine a bend and angle flow x minimizing the *total cost* $c(x) \stackrel{\text{def}}{=} \sum_{a \in A \cup B} c(a) \cdot x_a$ for some $\gamma > 0$.

Lemma 5.12 (Tamassia 1987) *There is a one-to-one correspondence between flows in $\mathcal{B}(G)$ and orthogonal shapes of G (up to the ordering of bends on each edge). Moreover, if $\gamma = 1$, the total cost of a bend and angle flow equals the number of bends in the orthogonal shape.*

Transformation of a bend and angle flow into an orthogonal shape is straightforward. Given a bend and angle flow x, simply let $s_e = \mathtt{l}^{x(f,g)}\mathtt{r}^{x(g,f)}$ for an edge e in the face list of a face f, where g is the face on the opposite side of e. Angles a_i between consecutive edge e_i and $e_{i+1 \mod d_G(f)}$ of a face

f are set to $x_{(v,f)} \cdot \frac{\pi}{2}$, where $(v, f) \in A$ is the arc entering f between these two edges.

To minimize the number of bends in an orthogonal representation of a plane 4-graph G it is hence sufficient to compute a minimum cost flow in $\mathcal{B}(G)$. Using minimum cost augmenting paths, this can be carried out in time $\mathcal{O}(n^2 \log n)$, where n is the number of vertices in G (Tamassia, 1987). By cleverly combining two minimum cost flow algorithms, Garg and Tamassia (1996) obtain the first subquadratic algorithm for bend minimization.

Theorem 5.13 (Garg and Tamassia 1995, 1996) *The shape of an orthogonal representation preserving the embedding of a plane 4-graph with the minimum number of bends can be determined in time $\mathcal{O}(n^{7/4} \log n)$. It is $\mathcal{NP}$-hard to minimize the number of bends over all embeddings.*

An algorithm minimizing the number of bends over all embeddings (with running time exponential in the number of vertices) is given by Bertolazzi *et al.* (1997). Di Battista *et al.* (1993b) show that the bend minimization problem can be solved efficiently for planar 3-graphs, and Didimo and Liotta (1998) fill in the gap with an algorithm for planar 4-graphs that is exponential only in the number of vertices with degree four.

5.2.2 Compaction

The final step of the Topology-Shape-Metrics approach is performed by embedding the orthogonal shape in a grid. The simple example in Figure 5.13 indicates why grid embeddings of the same orthogonal shape can look quite different, even if planarity is preserved. Besides planarity, the following are desirable features of a grid embedding (where distances are measured in terms of grid line segments):

- short total and maximum edge length
- small width, height, and area
- good aspect ratio

The problem of computing a grid embedding, *i.e.* integer coordinates for vertices and bends, while preserving planarity and shape is therefore called *compaction*. Recently, Patrignani (1999) showed that total edge length minimization, maximum edge length minimization, and area minimization are $\mathcal{NP}$-hard problems for general shapes, but Di Battista *et al.* (1999, Section 5.4) show that a grid embedding minimizing the total edge length, width,

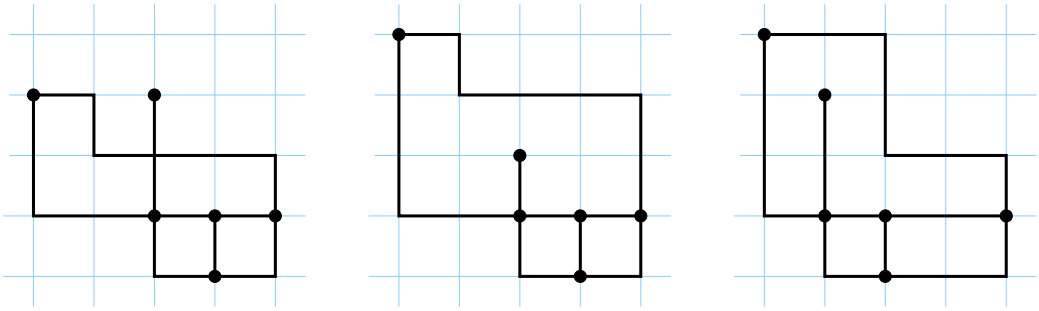


Figure 5.13: Three grid embeddings of the orthogonal shape from Figure 5.9

height, and area can be computed efficiently if the orthogonal shape represents every face as a rectangle. In this section, we indicate how an algorithm of Tamassia (1987) refines an orthogonal shape to have rectangular faces, and outline the optimizing compaction algorithm applicable to such shapes.

Tamassia (1987) refines orthogonal shapes to have rectangular faces by adding a number of dummy vertices and edges. The refined orthogonal shape is then embedded in the grid, and a grid embedding of the original graph is obtained simply by omitting all dummy vertices and edges.

Given a plane 4-graph G and an orthogonal shape $\mathcal{H}(G)$, a *rectangular refinement* of $\mathcal{H}(G)$ is an orthogonal shape $\mathcal{H}(G^{\square})$ of a graph $G^{\square}$ that has a dummy vertex for each bend of $\mathcal{H}(G)$, and some more dummy vertices and edges subdividing each face into rectangles. The restriction of $\mathcal{H}(G^{\square})$ to the elements of G must yield an orthogonal shape for G. See Figure 5.14.

Tamassia's algorithm now traverses each internal face of $G^{\square}$ in counterclockwise direction, connecting the vertex of each right turn with a dummy vertex subdiving the first edge that can oppose this turn. The external face

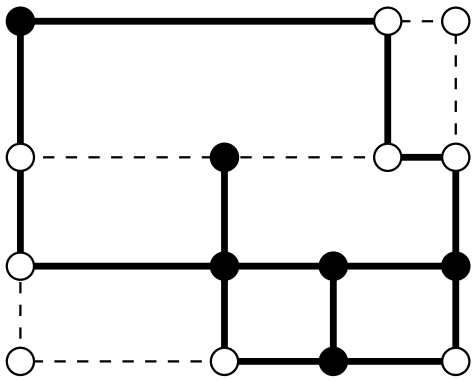


Figure 5.14: Rectangular refinement of the orthogonal shape from Figure 5.9

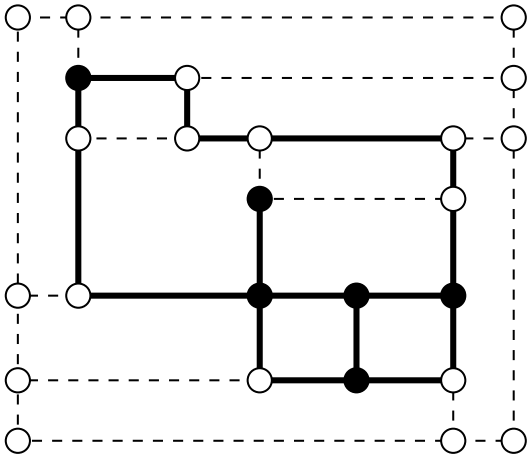


Figure 5.15: Rectangular refinement of the orthogonal shape from Figure 5.9 obtained from the algorithm of Tamassia (1987)

is traversed clockwise, and an enclosing rectangle of dummy vertices and edges is used to ensure that every right turn is matched with an edge. See Figure 5.15 and refer to Di Battista *et al.* (1999, Section 5.4.2) for details.

Given an orthogonal shape with rectangular faces for the plane 4-graph $G^{\square}$ we want to assign positive integer lengths to the edges. Feasible solutions are easily characterized.

Lemma 5.14 *A length assignment for the edges of an orthogonal shape with rectangular faces is consistent, if and only if, for every face, opposite sides have the same length.*

The edges of $G^{\square}$ can easily be classified into horizontal and vertical in one of two ways. By the above lemma, edge lengths can be determined independently for each class. Moreover, since the condition easily translates into flow conservation constraints, the total edge length minimization problem can be stated in terms of a minimum cost flow problem. Construction of the corresponding networks is shown in Figure 5.16. Using the minimum cost flow algorithm of Garg and Tamassia (1996), the following corollary is obtained.[4]

[4]Since there always is a flow in these compaction networks, this ultimately proves that every angle assignment corresponding to a bend and angle flow is planar realizable. It is an instructive exercise to relate the resulting grid embedding to the Ceva constraints introduced in Theorem 5.4.

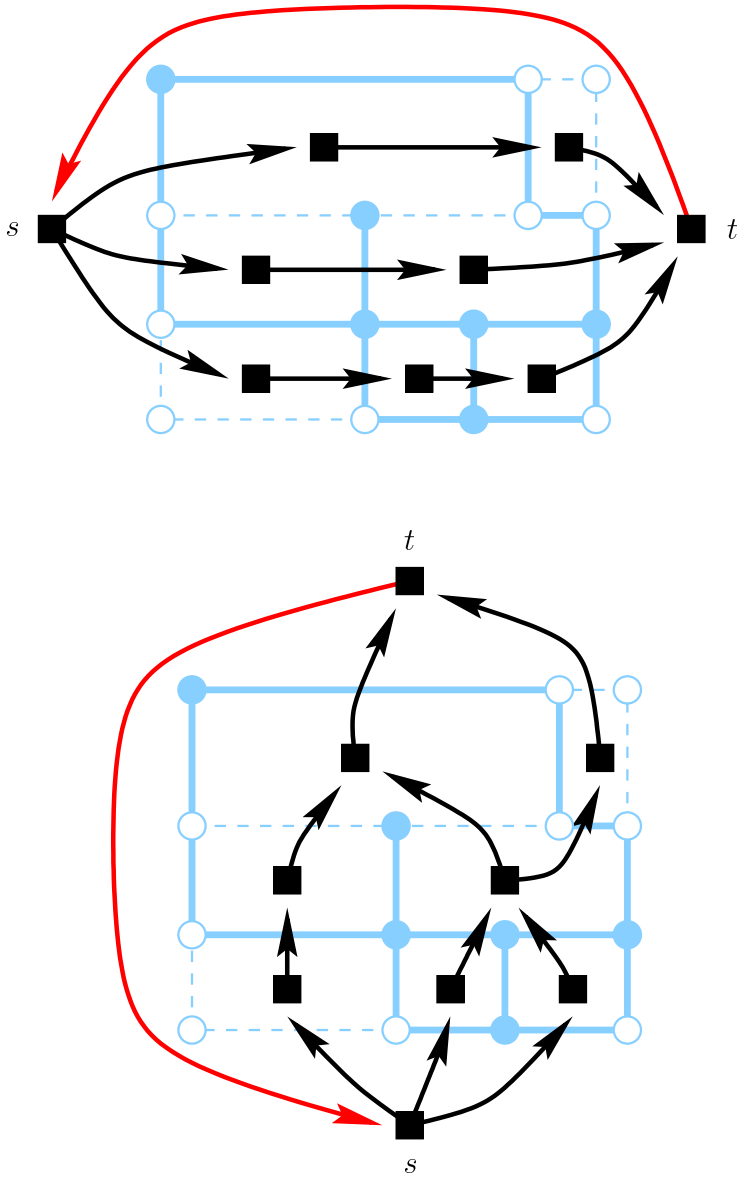


Figure 5.16: Compaction networks for horizontal and vertical edge segments. Except for arcs (t, s), which have lower capacity and cost zero, every arc has lower capacity one, infinite upper capacity and unit cost. Every node has demand/supply zero. Each unit of flow on an arc induces one grid segment for the edge crossed. Then, the sum of the total cost of both flows equals the total induced edge length

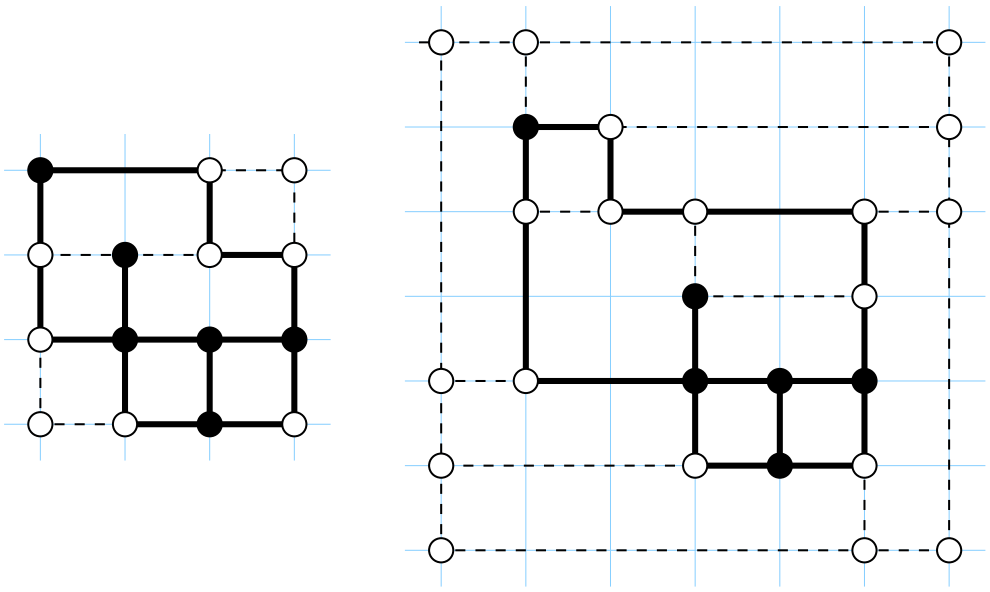


Figure 5.17: Optimal compactions of the rectangular refinements from Figures 5.14 (*left*) and 5.15 (*right*)

Corollary 5.15 *Given a plane* 4*-graph* $G^{\square}$ *and an orthogonal shape of* $G^{\square}$ *with rectangular faces, a shape preserving grid embedding with the minimum height, width, area, and total edge length can be computed in time* $\mathcal{O}\left(n^{7/4} \log n\right)$.

Tamassia (1987) cites a faster algorithm of Hsueh (1979, pp. 26–28/46–52) that is based on longest path computations in the dual graphs of the above compaction networks. However, this algorithm does not minimize the total edge length. A detailed description is given in Di Battista *et al.* (1999, Section 5.4.1).

Lemma 5.16 *Given a plane* 4*-graph* $G^{\square}$ *and an orthogonal shape of* $G^{\square}$ *with rectangular faces, a shape preserving grid embedding with the minimum height, width, and area can be computed in linear time.*

Figure 5.17 illustrates, why the total edge length of the resulting grid embedding of G is not necessarily minimum, even when the optimizing algorithm of Corollary 5.15 is used. It remains an open challenge to devise better rectangularization strategies.

5.3 Dynamic Orthogonal Shape

Orthogonal representations are popular especially for graphs in technical applications like network plans, Entity-Relationship diagrams, program dependencies, or circuit schematics. These settings often require graphs to be edited interactively, as in CASE tools or schema editors. Just like animation (see Section 3.3), user interaction is a natural source for dynamic graph layout problems.

As mentioned in Section 2.2, several approaches to dynamic layout have been taken (Cohen *et al.*, 1992; Böhringer and Paulisch, 1990; North, 1996a). In the realm of orthogonal representations, research has mostly focused on incremental updates, in which only creation of new vertices and edges is allowed (Papakostas and Tollis, 1996; Biedl and Kaufmann, 1997). In particular, Papakostas and Tollis (1996) discuss several scenarios with respect to the kind of modifications allowed in transition from one layout to the next: *Draw-From-Scratch*, *Full-Control*, *Relative-Coordinates*, and *No-Change*. *Draw-From-Scratch* and *No-Change* are at opposite ends of the trade-off between (static) layout quality and the attempt to preserve a user's mental map. In the *Full-Control* scenario, the user is allowed to request many features by means of constraints, while the main concern in the *Relative-Coordinates* scenario is to maintain the relative ordering of vertex positions along coordinate axes.

A fully dynamic system for orthogonal layout of dynamic 4-graphs is InteractiveGiotto (Bridgeman *et al.*, 1997). It is based on Giotto (Tamassia *et al.*, 1988), an implementation of the Topology-Shape-Metrics approach using the bend and angle network to determine the shape of a layout. InteractiveGiotto planarizes the graph according to the drawing currently visible by introducing new vertices for crossings and bends.[5] In the associated bend and angle network capacities of arcs stemming from unmodified portions of the graph are altered in order to fix the flow at its previous value. Hence, InteractiveGiotto does not change the shape of these parts of the graph, whereas new or modified portions are laid out bend-minimally. Note that the number of bends is never decreased, except when edges are deleted from the graph.

Here, we apply the Bayesian framework of Section 2.2 to obtain a dynamic model which forms an explicit and controllable compromise between layout stability and layout quality, *i.e.* between the modification of shape and the total number of bends. Our approach is described formally in the next section. The subsequent section shows how to determine optimal layouts in this

[5]The latter to maintain the ordering of bends on each edge.

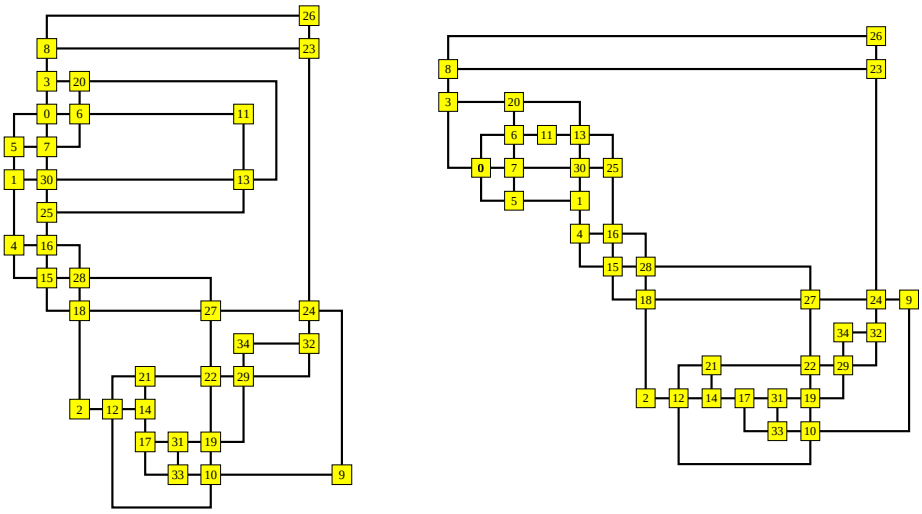


Figure 5.18: Two bend minimum layouts of a plane 4-graph

model efficiently. Interestingly, the corresponding optimization problem can still be solved by network flow techniques. Since an implementation of our model only realizes the shape step, it can be incorporated into any system for dynamic layout using the Topology-Shape-Metrics approach. Some issues related to integration in interactive systems are touched in Section 5.3.3.

5.3.1 Bend Number vs. Shape Modification

It was shown in Section 5.2.1 that bend and angle flows are well suited for bend minimization in orthogonal representations of static graphs. The main difficulty in extending the Topology-Shape-Metrics approach to dynamic graphs is rooted in the observation that two bend minimum layouts even of the same graph can have very different shapes (see Figure 5.18), not to mention pairs of layouts for graphs that are only similar. In an interactive graph layout tool, a *Draw-From-Scratch* scenario may thus result in consecutive drawings that do not seem related. The user's mental map is not retained at all.

On the other hand, a *No-Change* scenario results in drawings that do not satisfy the quality criterion that was the focus of the previous section, namely small bend numbers. Figure 5.19 gives an example. Moreover, an interactive application does not allow to maintain restrictive invariants necessary to preserve orthogonality and planarity throughout.

The other two scenarios of Papakostas and Tollis (1996) are difficult to adapt to our setting, because they are based on an actual grid embedding

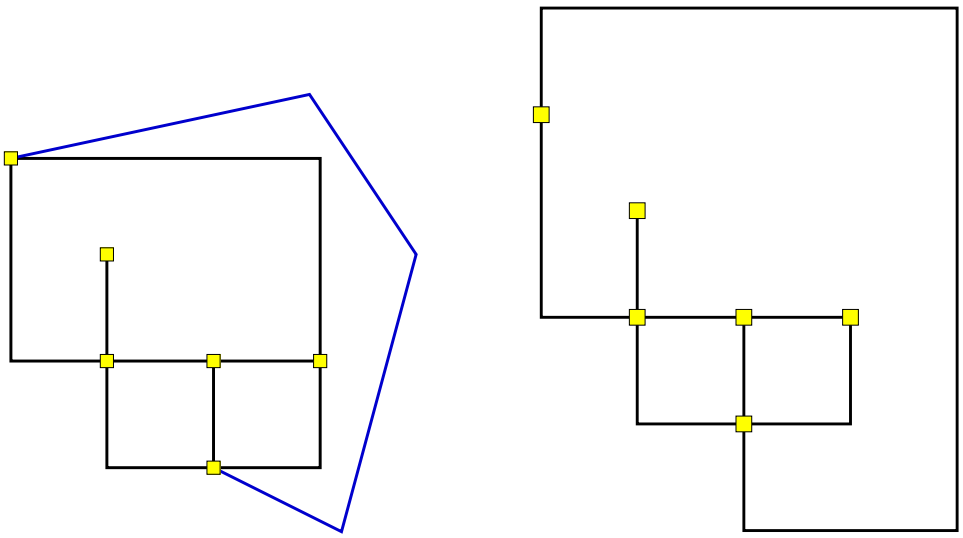


Figure 5.19: In the *No-Change* scenario, if feasible at all, modifications often result in orthogonal shapes with an unnecessary high number of bends. Here, creation and bend minimal routing of a new edge is followed by deletion of an old one

rather than the shape of an orthogonal representation. Even though InteractiveGiotto follows a *No-Change* scenario with respect to the orthogonal shape, the resulting layouts at least partially resemble a *Relative-Coordinates* scenario due to shifts caused by compaction.

The major drawback of InteractiveGiotto is the one common to any *No-Change* scenario. The quality of a layout is never actively improved, *i.e.* the number of bends is never decreased by the layout algorithm, but only by deletions. To avoid occasional recomputations of the whole layout as a work-around, possibly altering the shape substantially, we propose a different strategy based on the dynamic layout framework presented in Section 2.2. Just as in the case of dynamic straight-line embeddings (Section 3.3.1), we integrate the grade of change in values assigned to layout elements into the objective function.

According to the Bayesian framework of Section 2.2 stability can be incorporated into a dynamic model as a local criterion measuring the deviation from target values generated from the previous layout. In Section 3.3.1, the dependencies between consecutive layouts are modeled by target positions of vertices that are present in the layout prior to modification. The layout elements of a bend and angle network are the flow variables x_a, $a \in A \cup B$. We therefore assume that a set $\Lambda \subseteq A \cup B$ of arcs is given, for which target values are derived from the shape of a layout of the previous graph. Possible

ways of generating such values are outlined in the next section.

We hence assume that we are given a plane 4-graph $G = (V, E, F)$ with its associated bend and angle network $\mathcal{B}(G)$ and a vector y of target values for a subset Λ of arcs. The energy function of the static bend-minimum shape model is the cost $U(x) = \sum_{(f,g)\in B} \gamma \cdot x_{(f,g)}$ of flow across face-face arcs in $\mathcal{B}(G)$. By introducing stability potentials

$$U_{\{a\}}(x \mid y_a) = \mathsf{Deviation}\,(x_a, y_a \mid \rho_a) \stackrel{\text{def}}{=} \rho_a \cdot |x_a - y_a|$$

measuring the deviation from target values for all $a \in \Lambda$, we obtain a *dynamic orthogonal shape model*

$$U(x \mid y) = \gamma \cdot \sum_{(f,g)\in B} x_{(f,g)} + \alpha \cdot \sum_{(v,f)\in\Lambda\cap A} |x_{(v,f)} - y_{(v,f)}| + \beta \cdot \sum_{(f,g)\in\Lambda\cap B} |x_{(f,g)} - y_{(f,g)}|$$

Parameters α, β, and γ enable arbitrary relative weighting of all three criteria with integer values. Though this model was found independently, it can be considered a relaxation of the model underlying InteractiveGiotto. Instead of constraining some arcs to carry an amount of flow given by the shape of the previous layout, the absolute differences are to be minimized. The advantage of this approach is that the number of bends can be reduced, if the improvement outweights the number of changes to the orthogonal shape.

5.3.2 Compromise Optimization

Even though the dynamic orthogonal model is only piecewise linear, optimal shape can still be obtained using network flow techniques. The basic idea is to assume that each arc in the bend and angle network carries the targeted amount of flow, and to modify capacities as well as demand and supply values accordingly. Each unit of flow in the resulting network is charged its contribution to the objective function derived above. The remainder of this subsection gives the details of this construction.

Given any flow network $\mathcal{N} = (W, A; b, l, u, c)$ and a flow y, the *residual network (with respect to a flow y)*, $\mathcal{N}_y = (W, A \cup \bar{A}; b_y, l_y, u_y, c_y)$, is constructed by adding *reduction arcs* $\bar{a}$ for all $a \in A$, where $\bar{a}$ connects the same nodes as a, but is oriented in the opposite direction. The set of all reduction arcs is denoted by $\bar{A}$. Furthermore, we define

- residual supplies/demands $b_y(w) = 0$
- residual capacities
$$\begin{aligned} l_y(a) &= l_y(\bar{a}) = 0 \\ u_y(a) &= u(a) - y_a \\ u_y(\bar{a}) &= y_a - l(a) \end{aligned}$$

- residual costs $c_y(a) = c(a)$
  $c_y(\bar{a}) = -c(a)$

for all $w \in W$, $a \in A$. Every flow z in $\mathcal{N}_y$ yields a circulation $\chi(z)$ in $\mathcal{N}$ by setting

$$\chi(z)_a = z_a - z_{\bar{a}}$$

Clearly, $\chi(z)$ gives rise to a flow $x = y + \chi(z)$ in $\mathcal{N}$. This standard model for feasible altercation of flow (see, *e.g.*, Ahuja *et al.*, 1993) is modified in two aspects to obtain a network formulation for the optimization problem associated with our dynamic orthogonal shape model.

In principle, we have to account for both the facts that targets for flow values are given only for a subset of arcs, and that these values need not satisfy the capacity constraints. Without loss of generality we can however assume that all target values do satisfy the capacity constraints (otherwise replace y_a by $\min\{\, u(a), \max\{l(a), y_a\}\,\}$). We say that y_a is a *target flow value* for arc a to indicate that it satisfies the capacity constraints. For a subset $\Lambda \subseteq A$ we generalize the above definition to the *residual network (with respect to y defined on Λ)*, $\mathcal{N}_y = (W, A \cup \bar{A}; b_y, l_y, u_y, c_y)$, where

$$b_y(w) = b(w) + \sum_{(w',w)\in\Lambda} y_{(w',w)} - \sum_{(w,w')\in\Lambda} y_{(w,w')}$$

for all $w \in W$, and

$$\begin{aligned}
l_y(a) &= \begin{cases} 0 & \text{if } a \in \Lambda \\ l(a) & \text{otherwise} \end{cases} \\
l_y(\bar{a}) &= 0 \\
\\
u_y(a) &= \begin{cases} u(a) - y_a & \text{if } a \in \Lambda \\ u(a) & \text{otherwise} \end{cases} \\
u_y(\bar{a}) &= \begin{cases} y_a - l(a) & \text{if } a \in \Lambda \\ 0 & \text{otherwise} \end{cases} \\
\\
c_y(a) &= c(a) \\
c_y(\bar{a}) &= -c(a)
\end{aligned}$$

for all $a \in A$. Note that, although $\chi(z)$ of some flow z in $\mathcal{N}_y$ is no longer a circulation in $\mathcal{N}$, we still obtain a flow x in $\mathcal{N}$ by setting

$$x_a = \begin{cases} y_a + \chi_a(z) & \text{if } a \in \Lambda \\ \chi_a(z) & \text{otherwise} \end{cases}$$

For ease of notation we also write $x = y + \chi(z)$ to denote the flow thus obtained. The above case distinctions may seem overly complicated, because the same values for capacities, demands and supplies, and costs, are obtained by introducing new target values $y_a = 0$ for all $a \in A \setminus \Lambda$. We will see below, however, that there is a substantial difference between a target value of 0 and no target value at all.

The second modification is necessary because the dynamic orthogonal model, in addition to the usual cost of bends, penalizes deviation from target values. The flow values should deviate from their target flow values only if a notable improvement of the bend minimization cost function is achieved. Therefore, we render altercation of the assumed flow values more difficult by adding penalties ρ_a, $a \in \Lambda$, to residual costs. The resulting network is called the *penalized residual network*, $\mathcal{N}_y^\rho = (W, A \cup \bar{A}; b_y, l_y, u_y, c_y^\rho)$, where

$$c_y^\rho(a) = \begin{cases} c_y(a) + \rho_a & \text{if } a \in \Lambda \\ c_y(a) & \text{otherwise} \end{cases}$$

$$c_y^\rho(\bar{a}) = \begin{cases} c_y(\bar{a}) + \rho_a & \text{if } a \in \Lambda \\ c_y(\bar{a}) & \text{otherwise} \end{cases}$$

Penalties ρ_a represent the extra cost of each unit of flow deviating from the targeted amount along arcs $a \in \Lambda$. It therefore makes a difference, whether an arc has a target value of 0, or no target value at all (provided the penalty does not equal zero).

The following theorem summarizes the purpose of this construction. A flow z in the residual is called *proper*, if for all $a \in A$ at least one of z_a and $z_{\bar{a}}$ is zero.

Theorem 5.17 *Let y be a vector of target flow values for a set $\Lambda \subseteq A$ of arcs in a network $\mathcal{N}$. The total cost of a proper flow z in $\mathcal{N}_y^\rho$ equals*

$$\sum_{a \in A} c(a) \cdot x_a + \sum_{a \in \Lambda} \rho_a \cdot |x_a - y_a| - \sum_{a \in \Lambda} c(a) \cdot y_a$$

where $x = y + \chi(z)$ is a flow in $\mathcal{N}$.

■ **Proof** We first show that $x = y + \chi(z)$ is a flow in $\mathcal{N}$. To see that x satisfies the capacity constraints, first note that $z_a, z_{\bar{a}} \geq 0$ for all $a \in A$, since $l_y(a), l_y(\bar{a}) \geq 0$. If $a \in \Lambda$, then

$$l(a) = y_a - u_y(\bar{a}) \leq y_a - z_{\bar{a}} \leq x_a \leq y_a + z_a \leq y_a + u_y(a) = u(a)$$

If $a \notin \Lambda$, then $l_y(\bar{a}) = u_y(\bar{a}) = 0$ implies $z_{\bar{a}} = 0$, and consequently

$$l(a) = l_y(a) \leq z_a = x_a = z_a \leq u_y(a) = u(a)$$

Since z is a flow in $\mathcal{N}_y$, we also have

$$\begin{aligned}
b(w) &= b_y(w) - \sum_{(w',w)\in\Lambda} y_{(w',w)} + \sum_{(w,w')\in\Lambda} y_{(w,w')} \\
&= \sum_{(w,w')\in A} z_{(w,w')} - \sum_{(w',w)\in A} z_{(w',w)} - \sum_{(w,w')\in A} z_{\overline{(w,w')}} \\
&\quad + \sum_{(w',w)\in A} z_{\overline{(w',w)}} - \sum_{(w',w)\in\Lambda} y_{(w',w)} + \sum_{(w,w')\in\Lambda} y_{(w,w')} \\
&= \sum_{(w,w')\notin\Lambda} \chi_{(w,w')}(z) - \sum_{(w',w)\notin\Lambda} \chi_{(w',w)}(z) \\
&\quad + \sum_{(w,w')\in\Lambda} \big(y_{(w,w')} + \chi_{(w,w')}(z)\big) - \sum_{(w',w)\in\Lambda} \big(y_{(w',w)} + \chi_{(w',w)}(z)\big) \\
&= \sum_{(w,w')\in A} x_{(w,w')} - \sum_{(w',w)\in A} x_{(w',w)}
\end{aligned}$$

for all $w \in W$. Therefore, x is indeed a flow in $\mathcal{N}$.

Since for all $a \in A$ at least one of z_a and $z_{\bar{a}}$ is zero, the cost of z in $\mathcal{N}_y^\rho$ is

$$\begin{aligned}
&\sum_{a\in A} \big(c_y^\rho(a)\cdot z_a + c_y^\rho(\bar{a})\cdot z_{\bar{a}}\big) \\
&\qquad = \sum_{a\in A} c(a)\cdot(z_a - z_{\bar{a}}) + \sum_{a\in\Lambda} \rho_a\cdot(z_a + z_{\bar{a}}) \\
&\qquad = \sum_{a\notin\Lambda} c(a)\cdot x_a + \sum_{a\in\Lambda} c(a)\cdot(x_a - y_a) \\
&\qquad\quad + \sum_{a\in\Lambda\,:\,z_{\bar{a}}=0} \rho_a\cdot z_a + \sum_{a\in\Lambda\,:\,z_a=0} \rho_a\cdot z_{\bar{a}} \\
&\qquad = \sum_{a\notin\Lambda} c(a)\cdot x_a + \sum_{a\in\Lambda} c(a)\cdot(x_a - y_a) \\
&\qquad\quad + \sum_{a\in\Lambda\,:\,z_{\bar{a}}=0} \rho_a\cdot(x_a - y_a) + \sum_{a\in\Lambda\,:\,z_a=0} \rho_a\cdot(y_a - x_a) \\
&\qquad = \sum_{a\in A} c(a)\cdot x_a - \sum_{a\in\Lambda} c(a)\cdot y_a + \sum_{a\in\Lambda} \rho_a\cdot|x_a - y_a|
\end{aligned}$$

□

Application to our dynamic orthogonal shape model is straightforward. Simply let the weights of the stability potentials serve as penalties (cf. Figure 5.20).

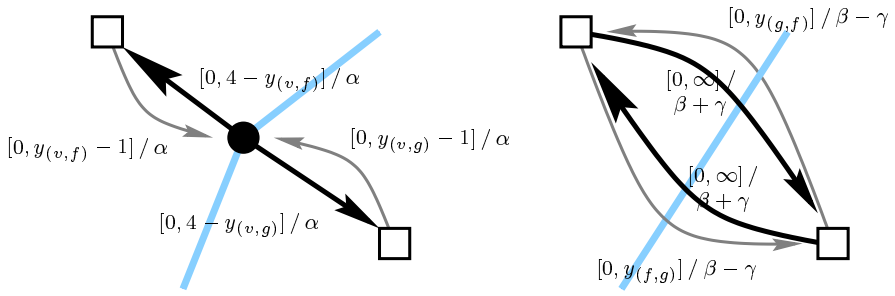


Figure 5.20: Reduction arcs in the residual bend and angle network with penalties from the dynamic orthogonal shape model

Corollary 5.18 *Given a plane 4-graph G and a vector y of target flow values defined on Λ, let $\rho = (\rho_a)_{a \in \Lambda}$ be defined by*

$$\rho_a = \begin{cases} \alpha & \text{if } a \in \Lambda \cap (V \times F) \\ \beta & \text{if } a \in \Lambda \cap (F \times F) \end{cases}$$

Then the cost of a proper flow z in $\mathcal{B}_y^\rho(G)$ equals $U(y + \chi(z) \mid y) - \gamma \cdot \sum_{a \in \Lambda \cap B} y_a$.

Using the minimum cost flow algorithm of Garg and Tamassia (1996) and the linear time compaction algorithm of Hsueh (1979), we can conclude that the dynamic orthogonal shape problem can be solved in subquadratic time for reasonable target flow values (*i.e.* implying a number of bends that is linear in the number of vertices).

Corollary 5.19 *Given a plane 4-graph G and a vector y of target flow values defined on Λ, an optimal layout according to the dynamic orthogonal shape model can be computed in time $\mathcal{O}\left(n(n+k)^{3/4} \log n + k\right)$, where n is the number of vertices in G, and $k = \sum_{a \in \Lambda \cap B} y_a$.*

It should go without notice that the concept of target values can be used in any flow based layout model. In particular, extensions to dynamic layout in the Kandinsky (Fößmeier and Kaufmann, 1996) and QuasiOrthogonal (Klau and Mutzel, 1998) models can be obtained along these lines.

Finally note that Fößmeier and Kaufmann (1997) use integer linear program formulations for advanced orthogonal shape models in order to avoid more complex network constructions. With the straightforward linear programming formulation of the penalized residual, any such model can be extended to dynamic layout by using target values. For each absolute difference $|x_a - y_a|$ in the objective function, new variables z_a and $z_{\bar{a}}$ represent the positive and negative difference, respectively. With the additional constraints

$z_a, z_{\bar{a}} \geq 0$ and $x_a - y_a = z_a - z_{\bar{a}}$, we can therefore substitute $|x_a - y_a|$ by $z_a + z_{\bar{a}}$ to obtain an equivalent linear program.

5.3.3 Interactive Systems

Dynamic orthogonal shape computation is but one step in interactive orthogonal graph layout following the Topology-Shape-Metrics approach. In this subsection, we briefly discuss possible ways of integration in a fully interactive environment.

Topology. At first sight, the topology step seems trivial, because an embedding is already given by the user of the interactive system. However, it cannot be assumed that it is planar at all times. To apply the network approach for angle assignment, the graph must hence be planarized. A common approach used in static settings is to compute a maximal planar subgraph and re-insert deleted edges into the final layout. This and related approaches are discussed by Jünger and Mutzel (1996). In an interactive tool, the planar subgraph approach cannot be expected to work particularly well, because it gives away control over the embedding, and hence does not preserve the user's mental map. A simple strategy that seems to be better suited for interactive environments is an online refinement of the embedded graph shown on the screen into a planar graph preserving the embedding. This is easily done by replacing each crossing by a new vertex of degree four and adding a vertex for each bend of the previous layout or drawn by the user (to maintain the order of left and right turns in the bend and angle network). See Figure 5.21.

Shape. Keeping track of those vertices and edges that have been modified after the most recent layout computation allows for a variety of target value generation schemes in the subsequent shape step. For each arc in the bend and angle network we can specify

- a fixed flow value (as in InteractiveGiotto)
- a target flow value together with a penalty for deviation (as described in the previous subsection)
- nothing at all (indicating that we do not care about shape modification)

A reasonable convention might be to

- let the user explicitly mark angles and bends that are not allowed to change

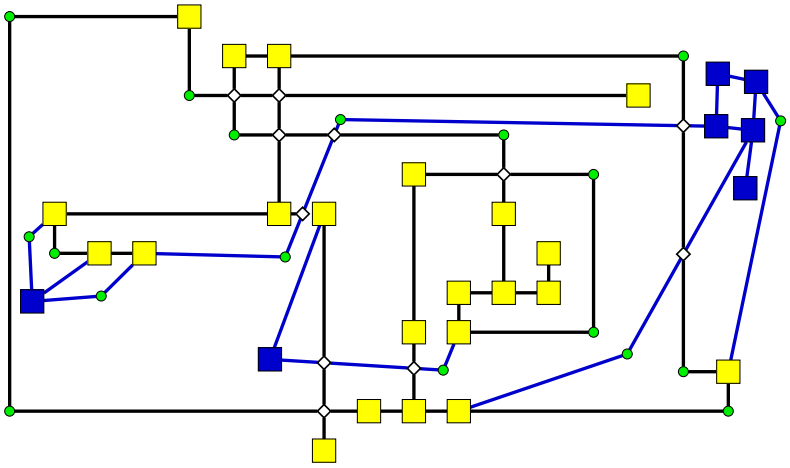


Figure 5.21: Planarization of an embedded graph. Compare with Figure 5.24(a) and observe that the order of edge segments at dummy vertices created by crossing edges prevents knock-knees in the orthogonal shape

- use angles and bends of unmodified portions of the graph from the previous layout as target values
- layout modified parts of the graph according to the bend-minimum model

Examples with target flow values for all angles and bends are given in Figures 5.22 and 5.23, while different schemes are compared in Figure 5.24, one of which is also applied in Figure 5.26.

Using a penalized residual network, the resulting energy minimization problem can be solved efficiently. Due to the fact that dummy vertices replace bends in the online planarization, we have to be a little more careful with the definition of the network. Figure 5.25 indicates the modification necessary to make sure that changing angles at these dummy vertices really has the effect of changing bends. Moreover, when a graph is not changed drastically between two layout computations, only few demand/supply values in the residual are different from zero. It can therefore be expected that the minimum cost flow problem is solved much faster than stated in Theorem 5.19 by computing (a small number of) augmenting paths.

To cope with vertices of degree higher than four, several variants based on the replacement of such vertices by *expansion cycles* exist (Tamassia *et al.*, 1988; Klau and Mutzel, 1998; Di Battista *et al.*, 1999, Section 5.8). An expansion cycle is a cycle of degree three vertices, each one connected to one of the former neighbors of the replaced vertex, while their cyclic order respects the

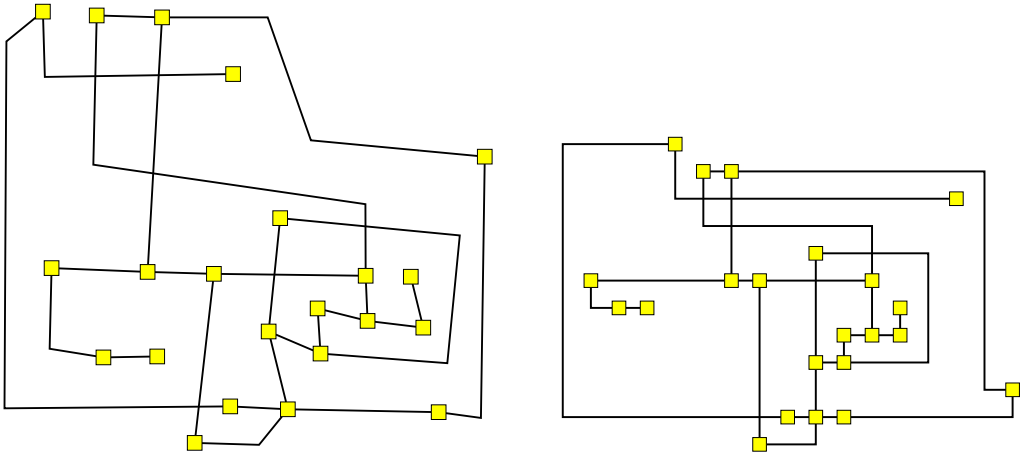


Figure 5.22: Orthogonalization of a hand-drawn sketch. All angles and bends in the drawing on the left are rounded to the next multiple of $\frac{\pi}{2}$ and used as target flow values for the dynamic model ($\alpha = \beta = \gamma = 1$)

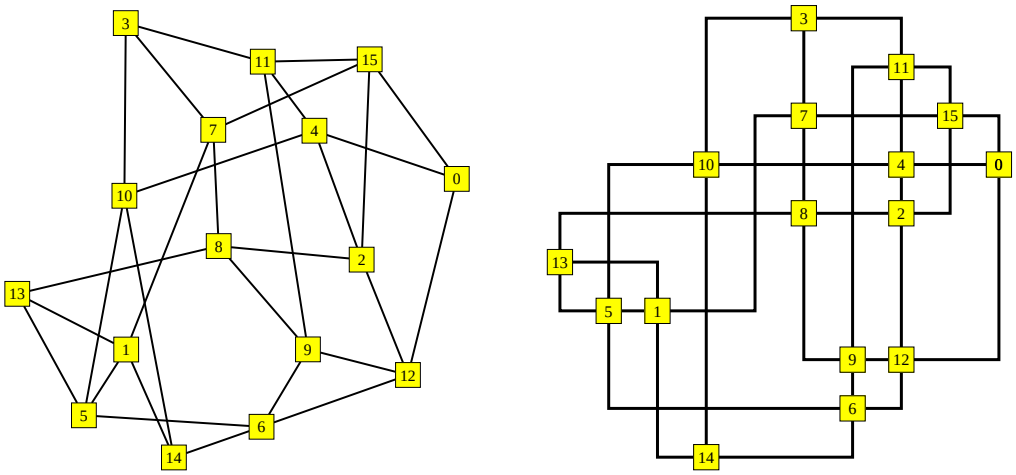
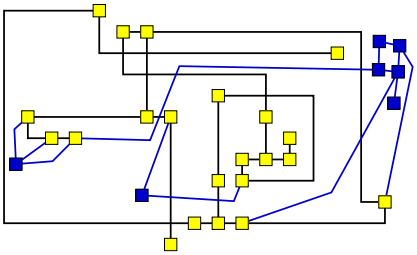
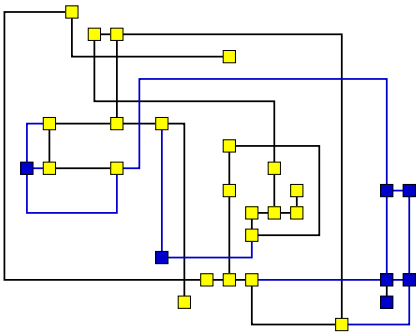


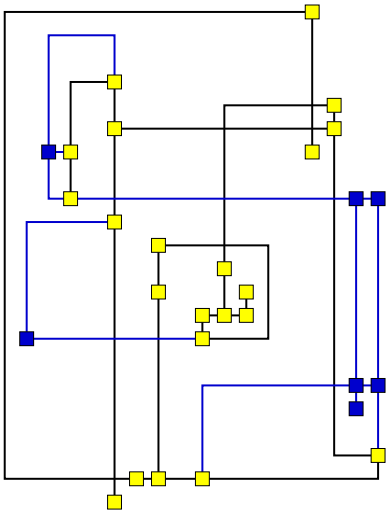
Figure 5.23: Orthogonalization of a spring embedder layout



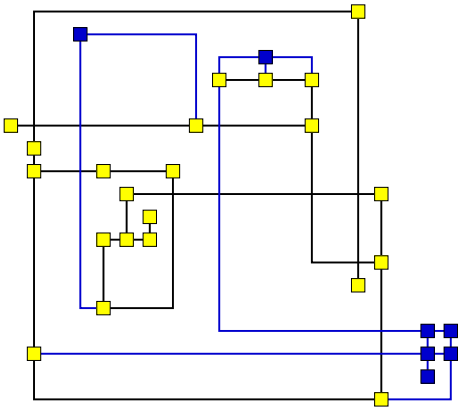
(a) The graph of Figure 5.22 with modifications



(b) Bias towards stability by generating target flow values for the whole graph ($\alpha = \beta = \gamma = 1$; 18 bends)



(c) No target values for modified parts ($\alpha = \beta = \gamma = 1$; 13 bends)



(d) No target flow values ($\alpha = \beta = 0$, $\gamma = 1$; 10 bends)

Figure 5.24: Selected strategies of target flow value generation

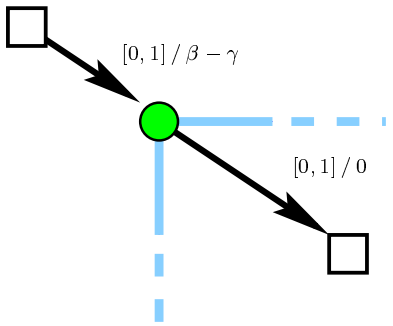


Figure 5.25: Network modification at bend replacing dummy vertices. These nodes have zero demand/supply and do not contribute to the demand values of incident faces

embedding. If all arcs in the bend and angle network corresponding to cycle edges are constrained to carry zero flow, the shape of each substituted cycle is guaranteed to be a rectangle, Vertices of the original graph can then be represented by rectangles (Tamassia *et al.*, 1988), or placed inside the rectangle (Klau and Mutzel, 1998). Quite differently, the KANDINSKY model allows zero degree angles between incident edges (Fößmeier and Kaufmann, 1996). The corresponding edge segments can thus run in parallel on a finer grid. Since zero degree angles are represented by further augmentation of the bend and angle network, all these approaches can be made dynamic using target values. However, finding intuitive schemes for target value generation becomes more difficult.

Metrics. Finally, a grid embedding of the orthogonal shape is to be determined. Since shape appears to dominate the impression of an orthogonal layout, static compaction as described in Section 5.2.2 may suffice.[6]

If the metrics step is based on the orthogonal shape alone, one occasionally encounters a rotation problem already mentioned by Bridgeman *et al.* (1997). Since there are two ways of classifying edges into vertical and horizontal, successive orthogonal representations may be rotated by 90 degrees. We suggest to use a majority vote of edge orientations from the representation visible prior to layout computation. Stability can be improved even further by using target values for length assignment, too.

[6]The reason for obvious detours in some of our examples is that we used an implementation of compaction based on rectangularization and the linear time algorithm of Hsueh (1979) (available in the AGD library, Mutzel *et al.*, 1998).

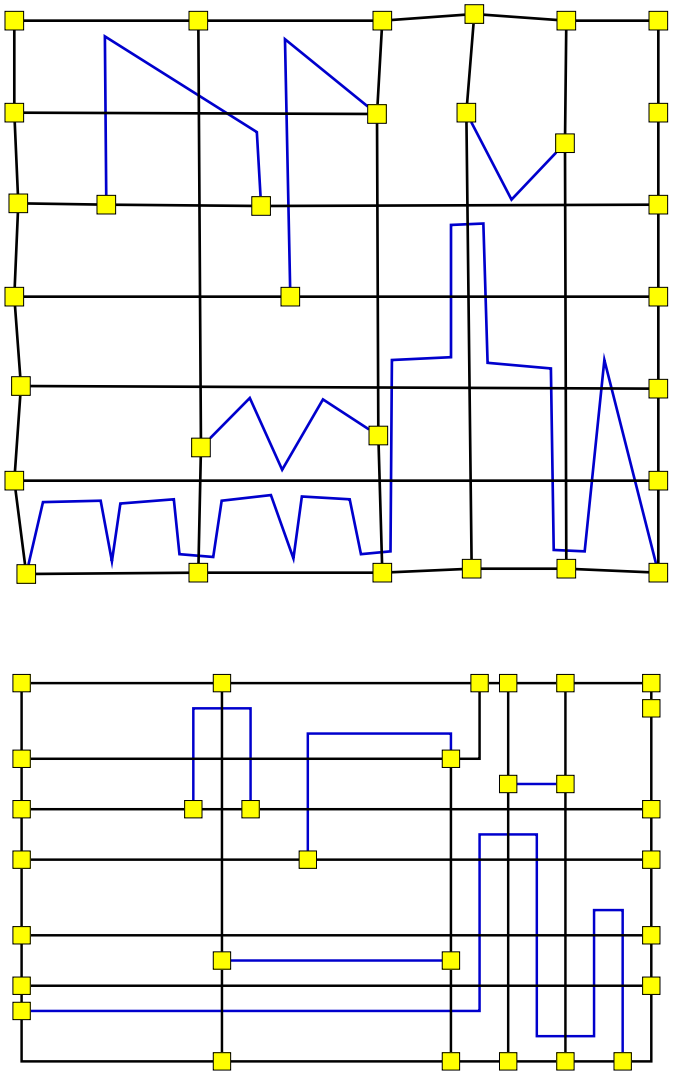


Figure 5.26: A sketched grid with subsequently added edges. Target flow values are generated to preserve the sketch, while edges added later are laid out bend minimally

Chapter 6

Conclusion

We broke down layout of graph visualizations and presented several new layout models and algorithms.

First, graph layout was embedded in the context of graphical presentation. For effective graphical presentation, substance must be conveyed in an easily comprehensible way. Because of the complex interdependencies involved in mapping structural to spatial relations, layout is the most challenging problem in the visualization of graphs. Grounded on formal descriptions of diagrams in general, we developed a uniform framework for graph layout, including layout of dynamic graphs. In this framework layout is considered an optimization problem with an objective function that separates the interdependencies into locally defined criteria of layout quality. A Bayesian argument then showed that dynamic graph layout can be considered a special case of static layout, where stability between layouts is modeled, *e.g.*, by local criteria measuring the deviation from target values.

At the cost of efficiency – a straightforward reduction shows that, in general, it is $\mathcal{NP}$-hard to obtain optimal layouts – the framework lends itself to generic implementation and therefore supports rapid prototyping of layout models. Note that satisfactory models can later be refined to become amenable to more efficient algorithms. Prototyping of layout models was exercised in Chapter 3 by means of case studies on graphs from different domains of application. The three case studies do not merely serve as proof of concept, but are interesting in their own right:

- For social networks we developed the first layout model that explicitly combines readability criteria with exact representation of substantive variables. Social networks pose many substantive problems, both quantitative and qualitative. It is our hope that our approach will initiate further research on the effective integration of clarity and substance.

- The application of our framework to an animation problem seamlessly extended static spring models to dynamic graphs. Since running time was dominated by viewpoint computation and rendering, we felt no need for faster layout algorithms. For interactive applications, however, this might well be an issue.

- In the case of time table graphs, we encountered some unintented beauty, because European public transportation networks represent some underlying geographical features amazingly well. Technically, the derived layout model appears to be the first force directed edge routing strategy. A better suited energy minimization method for the final model led to substantial speed up in layout computation.

As an example of energy based models that can be optimized efficiently, barycentric layout models were studied in Chapter 4. Using a construction of Eades and Garvan (1996), we gave simple proofs for the exponentially bad spatial and angular worst case resolution of these models. Since angles in barycentric layouts are an effect rather than an integral part of the model, this result bridges the gap to layouts methods that determine angles in the first place.

It is known that network flows capture some properties of angles in planar straight-line representations, but are insufficient to actually characterize them, at least in general. The latter was emphasized by proving the existence of locally consistent angle assignments for planar great-circle representations of triconnected planar graphs that display many desirable properties, including an optimal lower bound on the angular resolution, but are not realizable.

Angle networks do suffice, however, in the important special case that all angles are required to be multiples of $\frac{\pi}{2}$. The application of our framework for dynamic graph layout to the bend-minimum orthogonal shape approach of Tamassia (1987) led to the concept of target flow values. They were used to formulate various compromises between the number of bends and layout shape modifications in order to improve interactive orthogonal layout systems, but are likely to be useful in other network flow applications as well.

All proposed layout models have been implemented, and a number of example visualizations were shown.

One result of any learning experience is the ability to articulate questions. Consequently, new open problems arise from many topics touched in this thesis. Instead of an enumeration of selected problems, three general directions of further research that we consider relevant are given:

- For sure, interactive systems will become even more important. Similar to word processors with online formatting capabilities, dynamic graph layout could transparently support users in editing relational structures (*e.g.* in visual programming or system design). Layout in such systems has to be fast and flexible, implying the need for layout models with meaningful options as well as efficient algorithms. We used target values for stability in two otherwise quite different layout models. Do they yield extensions of other static models, or are there other useful instantiations of the Bayesian framework?

- Recently initiated work on difference metrics (Bridgeman and Tamassia, 1998) may serve as a basis for experimental studies evaluating different approaches for dynamic layout. But assessment of the effectiveness of graphical presentations is a pressing topic for graph drawing in general. While it is known that layout does have a strong effect on the ease and correctness of understanding structural information, few results indicate how layout can actively enhance understanding. The only experiments we know of suggest that spatial proximity causes viewers to group vertices, even if these do not form a dense subgraph (McGrath *et al.*, 1998). Besides it being plausible, we have no further evidence, for instance, that matching structural centrality in social networks with its geometric intuition is really effective.

- Syntactic (like centrality) and semantic (like non-structural grouping) substance obviously varies across applications. The linkage between clarity and domain specific substance will undoubtedly be the richest source of new graph drawing problems.

Bibliography

Aarts, E. H. L. and Korst, J. H. M. (1989). *Simulated Annealing and Boltzmann Machines*. Wiley.

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1993). *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall.

Azencott, R. (1990). Synchronous Boltzmann machines and Gibbs fields: Learning algorithms. In Foglman Soulie, F., editor, *Neurocomputing*, volume 68 of *NATO ASI Series F*, pages 51–62. Springer.

Batagelj, V. and Mrvar, A. (1998). PAJEK – Program for large network analysis. *Connections*, 21:47–57. Project home page at `http://vlado.mat.uni-lj.si/pub/networks/pajek/default.htm`.

Berkowitz, S. D. (1982). *An Introduction to Structural Analysis: The Network Approach to Scocial Research*. Butterworths.

Bertin, J. (1983). *Semiology of Graphics: Diagrams, Networks, Maps*. University of Wisconsin Press.

Bertolazzi, P. and Di Battista, G. (1991). On upward drawing testing of triconnected digraphs. In *Proceedings of the 7th ACM Annual Symposium on Computational Geometry (SoCG '91)*, pages 272–280. ACM, The Association for Computing Machinery.

Bertolazzi, P., Di Battista, G., and Didimo, W. (1997). Computing orthogonal drawings with the minimum number of bends. In *Proceedings of the Workshop on Algorithms and Data Structures (WADS '97)*, volume 1272 of *Lecture Notes in Computer Science*, pages 331–344. Springer.

Bertolazzi, P., Di Battista, G., and Liotta, G. (1995). Parametric graph drawing. *IEEE Transactions on Software Engineering*, 21(8):662–673.

Besag, J. (1986). On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B*, 48(3):259–302.

Bézier, P. (1972). *Numerical Control*. Wiley.

Biedl, T. C. and Kaufmann, M. (1997). Area-efficient static and incremental graph drawings. In Burkhard, R. and Woeginger, G., editors, *Proceedings of the 5th European Symposium on Algorithms (ESA '97)*, volume 1284 of *Lecture Notes in Computer Science*, pages 37–52. Springer.

Blythe, J., McGrath, C., and Krackhardt, D. (1996). The effect of graph layout on inference from social network data. In Brandenburg (1996), pages 40–51.

Böhringer, K.-F. and Paulisch, F. N. (1990). Using constraints to achieve stability in automatic graph layout algorithms. In *Proceedings of the ACM Human Factors in Computing Systems Conference (CHI '90)*, pages 43–51. ACM, The Association for Computing Machinery.

Bose, P., Gomez, F., Ramos, P., and Toussaint, G. (1996). Drawings nice projections of objects in space. In Brandenburg (1996), pages 52–63.

Brandenburg, F. J., editor (1996). *Proceedings of the 3rd International Symposium on Graph Drawing (GD '95)*, volume 1027 of *Lecture Notes in Computer Science*. Springer.

Brandenburg, F. J., Himsolt, M., and Rohrer, C. (1996). An experimental comparison of force-directed and randomized graph drawing algorithms. In Brandenburg (1996), pages 76–87.

Brandenburg, F. J., Jünger, M., and Mutzel, P. (1997). Algorithmen zum automatischen Zeichnen von Graphen. *Informatik-Spektrum*, 20:199–207.

Brandes, U., Kenis, P., Raab, J., Schneider, V., and Wagner, D. (1999). Explorations into the visualization of policy networks. *Journal of Theoretical Politics*, 11(1):75–106.

Brandes, U. and Wagner, D. (1997). A Bayesian paradigm for dynamic graph layout. In Di Battista (1997), pages 236–247.

Brandes, U. and Wagner, D. (1998a). Dynamic grid embedding with few bends and changes. In Chwa and Ibarra (1998), pages 89–98.

Brandes, U. and Wagner, D. (1998b). Using graph layout to visualize train interconnection data. In Whitesides (1998), pages 44–56.

Brandes, U. and Wagner, D. (1999). Über das Zeichnen von Graphen. In Horster, P., editor, *Festschrift zu Ehren von Walter Oberschelp*. Vieweg. In preparation.

Bridgeman, S., Fanto, J., Garg, A., Tamassia, R., and Vismara, L. (1997). InteractiveGiotto: An algorithm for interactive orthogonal graph drawing. In Di Battista (1997), pages 303–308.

Bridgeman, S., Garg, A., and Tamassia, R. (1996). A graph drawing and translation service on the WWW. In North (1996b), pages 45–52. Project home page at `http://loki.cs.brown.edu:8081/graphserver/`.

Bridgeman, S. and Tamassia, R. (1998). Difference metrics for interactive orthogonal graph drawing algorithms. In Whitesides (1998), pages 57–71.

Brightwell, G. R. and Scheinerman, E. R. (1993). Representations of planar graphs. *SIAM Journal on Discrete Mathematics*, 6(2):214–229.

Brooks, R. L., Smith, C. A. B., Stone, A. H., and Tutte, W. T. (1940). The dissection of rectangles into squares. *Duke Mathematical Journal*, 7:312–340.

Bruß, I. and Frick, A. (1996). Fast interactive 3-D graph visualization. In Brandenburg (1996), pages 99–110.

Burt, R. S. (1982). *Toward a Structural Theory of Action: Network Models of Social Structure, Perception, and Action*. Academic Press.

Chaiken, S. and Kleitman, D. (1978). Matrix tree theorems. *Journal of Combinatorial Theory, Series A*, 24:377–381.

Chen, M.-H. and Schmeiser, B. (1993). Performance of the Gibbs, Hit-and-Run, and Metropolis samplers. *Journal of Computational and Graphical Statistics*, 2(3):251–272.

Chwa, K.-Y. and Ibarra, O. H., editors (1998). *Proceedings of the 9th Annual International Symposium on Algorithms and Computation (ISAAC '98)*, volume 1533 of *Lecture Notes in Computer Science*. Springer.

Cleveland, W. S. (1985). *The Elements of Graphing Data*. Wadsworth Advanced Books and Software.

Cleveland, W. S. (1993). *Visualizing Data*. Hobart Press.

Cleveland, W. S. and McGill, R. (1984). Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association*, 79(387):531–554.

Cohen, R. F., Di Battista, G., Tamassia, R., and Tollis, I. G. (1995). Dynamic graph drawings: Trees, series-parallel digraphs, and planar *st*-digraphs. *SIAM Journal on Computing*, 24(5):970–1001.

Cohen, R. F., Di Battista, G., Tamassia, R., Tollis, I. G., and Bertolazzi, P. (1992). A framework for dynamic graph drawing. In *Proceedings of the 8th ACM Annual Symposium on Computational Geometry (SoCG '92)*, pages 261–270. ACM, The Association for Computing Machinery.

Conran, T. (1996). *Design*. Conran Octopus.

Cruz, I. F. and Tamassia, R. (1994). How to visualize a graph: Specification and algorithms. Tutorial on graph drawing, available at `http://ftp.cs.brown.ed/calendar/gd94/tutorial.html`.

Cruz, I. F. and Twarog, J. P. (1996). 3D graph drawing with simulated annealing. In Brandenburg (1996), pages 162–165.

Davidson, R. and Harel, D. (1996). Drawing graphs nicely using simulated annealing. *ACM Transactions on Graphics*, 15(4):301–331.

de Fraysseix, H., Pach, J., and Pollack, R. (1990). How to draw a planar graph on a grid. *Combinatorica*, 10:41–51.

Di Battista, G., editor (1997). *Proceedings of the 5th International Symposium on Graph Drawing (GD '97)*, volume 1353 of *Lecture Notes in Computer Science*. Springer.

Di Battista, G., Eades, P., de Fraysseix, H., Rosenstiehl, P., and Tamassia, R., editors (1993a). *Proceedings of the ALCOM International Workshop on Graph Drawing (GD '93)*. Available at `ftp://ftp.cs.brown.edu/pub/gd94/gd-92-93/gd93-v2.ps.Z`.

Di Battista, G., Eades, P., Tamassia, R., and Tollis, I. G. (1994). Algorithms for drawing graphs: An annotated bibliography. *Computational Geometry*, 4:235–282.

Di Battista, G., Eades, P., Tamassia, R., and Tollis, I. G. (1999). *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall.

Di Battista, G., Liotta, G., and Vargiu, F. (1993b). Spirality of orthogonal representations and optimal drawings of series-parallel graphs and 3-planar graphs. In *Proceedings of the Workshop on Algorithms and Data Structures (WADS '93)*, volume 709, pages 151–162. Springer.

Di Battista, G. and Vismara, L. (1996). Angles of planar triangulated graphs. *SIAM Journal on Discrete Mathematics*, 9(3):349–359.

Didimo, W. and Liotta, G. (1998). Computing orthogonal drawings in a variable embedding setting. In Chwa and Ibarra (1998), pages 79–88.

Ding, C. and Mateti, P. (1990). A framework for the automated drawing of data structure diagrams. *IEEE Transactions on Software Engineering*, 16(5):543–557.

Dobkin, D. P., Gansner, E. R., Koutsofios, E., and North, S. C. (1997). Implementing a general-purpose edge router. In Di Battista (1997), pages 262–271.

Doreian, P. (1988). Using multiple network analytic tools for a single social network. *Social Networks*, 10:287–312.

Eades, P. (1984). A heuristic for graph drawing. *Congressus Numerantium*, 42:149–160.

Eades, P. and Garvan, P. (1996). Drawing stressed planar graphs in three dimensions. In Brandenburg (1996), pages 212–223.

Eades, P., Lai, W., Misue, K., and Sugiyama, K. (1991). Preserving the mental map of a diagram. In *Proceedings of Compugraphics '91*, pages 24–33.

Eades, P. and Marks, J. (1995). Graph drawing contest report. In Tamassia and Tollis (1995), pages 143–146.

Eades, P. and Marks, J. (1996). Graph-drawing contest report. In Brandenburg (1996), pages 224–233.

Eades, P., Marks, J., Mutzel, P., and North, S. C. (1998). Graph drawing contest report. In Whitesides (1998), pages 423–435.

Eades, P., Marks, J., and North, S. C. (1996). Graph-drawing contest report. In North (1996b), pages 129–138.

Eades, P., Marks, J., and North, S. C. (1997). Graph-drawing contest report. In Di Battista (1997), pages 438–445.

Eick, S. G. (1996). Aspects of network visualization. *IEEE Computer Graphics and Applications*, 16(2):69–72.

Fáry, I. (1948). On straight-line representation of planar graphs. *Acta Scientiarum Mathematicarum (Institutum Bolyainum, Universitatis Szegediensis)*, 11:229–233.

Fisk, C. J., Caskey, D. L., and West, L. E. (1967). ACCEL: Automated circuit card etching layout. *Proceedings of the IEEE*, 55(11):1971–1982.

Formann, M., Hagerup, T., Haralambides, J., Kaufmann, M., Leighton, F. T., Symvonis, A., Welzl, E., and Woeginger, G. (1993). Drawing graphs in the plane with high resolution. *SIAM Journal on Computing*, 22(5):1035–1052.

Fößmeier, U. and Kaufmann, M. (1996). Drawing high degree graphs with low bend numbers. In Brandenburg (1996), pages 254–266.

Fößmeier, U. and Kaufmann, M. (1997). Algorithms and area bounds for nonplanar orthogonal drawings. In Di Battista (1997), pages 134–145.

Freeman, L. C. (1977). A set of measures of centrality based on betweenness. *Sociometry*, 40:35–41.

Freeman, L. C. (1979). Centrality in social networks: Conceptual clarification. *Social Networks*, 1:215–239.

Freeman, L. C. (1996). Visualizing social networks. Manuscript, available at `http://carnap.ss.uci.edu/vis.html`.

Freeman, L. C. (1997). Using available graph theoretic or molecular modeling programs in social network analysis. Manuscript, available at `http://tarski.ss.uci.edu/new.html`.

Frick, A., Ludwig, A., and Mehldau, H. (1995). A fast adaptive layout algorithm for undirected graphs. In Tamassia and Tollis (1995), pages 388–403.

Fröhlich, M. and Werner, M. (1995). Demonstration of the interactive graph visualization system daVinci. In Tamassia and Tollis (1995), pages 266–269. Project home page at `http://www.informatik.uni-bremen.de/~inform/forschung/daVinci/daVinci.h%tml`.

Fruchterman, T. M. and Reingold, E. M. (1991). Graph-drawing by force-directed placement. *Software—Practice and Experience*, 21(11):1129–1164.

Gansner, E. R. and North, S. C. (1998). Improved force-directed layouts. In Whitesides (1998), pages 364–373.

Garey, M. R. and Johnson, D. S. (1991). *Computers and Intractability: A Guide to the Theory of $\mathcal{NP}$–Completeness.* W.H. Freeman & Co.

Garg, A. (1995). On drawing angle graphs. In Tamassia and Tollis (1995), pages 84–95.

Garg, A. (1996). *Where to Draw the Line.* PhD thesis, Brown University.

Garg, A. and Tamassia, R. (1994). Planar drawings and angular resolution: Algorithms and bounds. In van Leeuwen, J., editor, *Proceedings of the 2nd European Symposium on Algorithms (ESA '94)*, volume 855 of *Lecture Notes in Computer Science*, pages 12–23. Springer.

Garg, A. and Tamassia, R. (1995). On the complexity of upward and rectilinear planarity testing. In Tamassia and Tollis (1995), pages 286–297.

Garg, A. and Tamassia, R. (1996). A new minimum cost flow algorithm with applications to graph drawing. In North (1996b), pages 201–216.

Garland, K. (1994). *Mr. Beck's Underground Map.* Capital Transport Publishers.

Garvey, R. B. (1998). Multidimensional outlines – Wordgraphs™. In Whitesides (1998), pages 448–449.

Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Matching and Machine Intelligence*, 6(6):721–741.

Geman, S., Geman, D., Graffigne, C., and Dong, P. (1990). Boundary detection by constrained optimization. *IEEE Transactions on Pattern Matching and Machine Intelligence*, 12:609–628.

Goldschmidt, O. and Takvorian, A. (1994). An efficient graph planarization two-phase heuristic. *NETWORKS*, 24:69–73.

Griffeath, D. (1976). Introduction to random fields. In Kemeny, J. G., Snell, J. L., and Knapp, A. W., editors, *Denumerable Markov Chains*, chapter 12, pages 425–458. Springer.

Guyon, X. (1995). *Random Fields on a Network*. Springer.

Hage, P. and Harary, F. (1995). Eccentricity and centrality in networks. *Social Networks*, 17:57–63.

Himsolt, M. (1996). The GRAPHLET system. In North (1996b), pages 233–240. Project home page at `http://www.fmi.uni-passau.de/Graphlet/`.

Hsueh, M.-Y. (1979). *Symbolic Layout and Compaction of Integrated Circuits*. PhD thesis, University of California, Berkeley.

Hutschenreuther, H. (1967). Einfacher Beweis des Matrix-Gerüst-Satzes der Netzwerktheorie. *Wissenschaftliche Zeitschrift, Technische Hochschule Ilmenau*, 13:403–404.

Jick, T. D. (1979). Mixing qualitative and quantitative methods: Triangulation in action. *Administrative Science Quarterly*, 24:602–611.

Johnson, D. S. (1982). The $\mathcal{NP}$-completeness column: An ongoing guide. *Journal of Algorithms*, 3:89–99.

Jünger, M. and Mutzel, P. (1996). Maximum planar subgraphs and nice embeddings: Practical layout tools. *Algorithmica*, 16:33–59.

Kamada, T. and Kawai, S. (1988). A simple method for computing general positions in displaying three-dimensional objects. *Computer Vision, Graphics and Image Processing*, 41:43–56.

Kamada, T. and Kawai, S. (1989). An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31:7–15.

Kamps, T., Kleinz, J., and Read, J. (1996). Constraint-based spring-model algorithms for graph layout. In Brandenburg (1996), pages 349–360.

Kant, G. (1993). *Algorithms for Drawing Planar Graphs*. PhD thesis, Utrecht University.

Kant, G. (1996). Drawing planar graphs using the canonical ordering. *Algorithmica*, 16:4–32.

Kenis, P. (1999). Analysing social network data by means of visualisation techniques. Presentation at the *19th International Conference on Social Network Analysis (Sunbelt XIX)*, Charleston.

Kenis, P. and Schneider, V. (1991). Policy networks and policy analysis: Scrutinizing a new analytical toolbox. In Marin, B. and Mayntz, R., editors, *Policy Networks: Empirical Evidence and Theoretical Considerations*, pages 25–59. Campus Verlag.

Kirchhoff, G. (1847). Über die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Verteilung galvanischer Ströme geführt wird. *Annalen der Physik und Chemie*, 72:497–508.

Kirckpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Klau, G. W. and Mutzel, P. (1998). Quasi-orthogonal drawing of planar graphs. Technical Report 98-1-013, Max-Planck-Institut für Informatik, Saarbrücken.

Klovdahl, A. S. (1981). A note on images of networks. *Social Networks*, 3:197–214.

Krackhardt, D., Blythe, J., and McGrath, C. (1994). KrackPlot 3.0: An improved network drawing program. *Connections*, 17(2):53–55.

Krempel, L. (1993). Simple representation of complex networks: Strategies for visualizing network structure. Paper presented at the *3rd European Conference for Social Network Analysis*, München. Available at `ftp://ftp.mpi-fg-koeln.mpg.de/pub/people/lk/simple.zip`.

Krempel, L. (1997). A gallery of social structures. Web presentation at `http://www.mpi-fg-koeln.de/~lk/netvis.html`.

Kriesi, H. (1982). The structure of the Swiss political system. In Lehmbruch, G., editor, *Patterns of Corporatist Policy-Making*, pages 133–163. Sage Publications.

Kruskal, J. B. and Wish, M. (1978). *Multidimensional Scaling*, volume 07-011 of *Sage University Paper series on Quantitative Applications in the Social Sciences*. Sage Publications.

Lauer, H., Ettrich, M., and Soukup, K. (1997). GraVis – System demonstration. In Di Battista (1997), pages 344–349. Project home page at `http://www-pr.informatik.uni-tuebingen.de/GraVis/index.html/`.

Laumann, E. O. and Knoke, D. (1987). *The Organizational State: Social Choice in National Policy Domains*. University of Wisconsin Press.

Li, S. Z. (1995). *Markov Random Field Modeling in Computer Vision.* Springer.

Lipton, R., Rose, D., and Tarjan, R. E. (1979). Generalized nested dissection. *SIAM Journal on Numerical Analysis*, 16:346–358.

Lüders, P., Ernst, R., and Stille, S. (1995). An approach to automatic display layout using combinatorial optimization algorithms. *Software—Practice and Experience*, 25(11):1183–1202.

Lundberg, G. A. and Steele, M. (1938). Social attraction patterns in a village. *Sociometry*, 1:327–419.

Lyons, K. A., Meijer, H., and Rappaport, D. (1998). Algorithms for cluster busting in anchored graph drawing. *Journal on Graph Algorithms and Applications*, 2(1):1–24.

Mackinlay, J. D. (1986). Automating the design of graphical presentations of relational data. *ACM Transactions on Graphics*, 5(2):110–141.

Mahon, B. H. (1977). Statistics and decision: The importance of communication and the power of graphical presentation. *Journal of the Royal Statistical Society, Series A*, 140(3):298–323.

Malitz, S. and Papakostas, A. (1994). On the angular resolution of planar graphs. *SIAM Journal on Discrete Mathematics*, 7(2):172–183.

Marks, J. (1991). A formal specification scheme for network diagrams that facilitates automated design. *Journal on Visual Languages and Computing*, 2:395–414.

Masui, T. (1994). Evolutionary learning of graph layout constraints from examples. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '94)*, pages 103–108. ACM, The Association for Computing Machinery.

Maxwell, J. C. (1874). On the application of Kirchhoff's rules for electric circuits to the solution of a geometrical problem. *Nature*, 10:411.

Mayntz, R. (1994). *Deutsche Forschung im Einigungsprozeß.* Campus Verlag.

McDonald, J. A. and Pedersen, J. (1991). Geometric abstractions for constrained optimization of layouts. In Buja, A. and Tukey, P. A., editors, *Computing and Graphics in Statistics*, volume 36 of *The IMA Volumes in Mathematics and its Applications*, pages 95–105. Springer.

McGrath, C., Blythe, J., and Krackhardt, D. (1997). The effect of spatial arrangement on judgments and errors in interpreting graphs. *Social Networks*, 19(3):223–242.

McGrath, C., Blythe, J., and Krackhardt, D. (1998). Seeing groups in graph layouts. Manuscript, available at `http://www.andrew.cmu.edu/user/cm3t/groups.html`.

Mehlhorn, K. and Näher, S. (1999). *The* LEDA *Platform of Combinatorial and Geometric Computing*. Cambridge University Press. Project home page at `http://www.mpi-sb.mpg.de/LEDA/`.

Mendonça, X. and Eades, P. (1993). Learning aesthetics for visualization. In *Anais do XX Seminário Integrado de Software e Hardware*, pages 76–88, Florianópolis, Brazil.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092.

Mohar, B. (1991). The Laplacian spectrum of graphs. In Alavi, Y., editor, *Proceedings of the 2nd International Conference on Graph Theory, Combinatorics, Algorithms, and Applications*, pages 871–898. SIAM.

Moreno, J. L. (1953). *Who Shall Survive: Foundations of Sociometry, Group Psychotherapy, and Sociodrama*. Beacon House.

Müller, K. H. (1991). *Symbole, Statistik, Computer, Design*. Verlag Hölder-Pichler-Tempsky.

Mutzel, P., Gutwenger, C., Brockenauer, R., Fialko, S., Klau, G. W., Krüger, M., Ziegler, T., Näher, S., Alberts, D., Ambras, D., Koch, G., Jünger, M., Buchheim, C., and Leipert, S. (1998). A library of algorithms for graph drawing. In Whitesides (1998), pages 456–457. Project home page at `http://www.mpi-sb.mpg.de/AGD/`.

North, S. C. (1996a). Incremental layout with DynaDag. In Brandenburg (1996), pages 409–418.

North, S. C., editor (1996b). *Proceedings of the 4th International Symposium on Graph Drawing (GD '96)*, volume 1190 of *Lecture Notes in Computer Science*. Springer.

Northway, M. L. (1940). A method for depicting social relationships obtained by sociometric testing. *Sociometry*, 3:144–150.

Padgett, J. F. and Ansell, C. K. (1993). Robust action and the rise of the Medici. 1400–1434. *American Journal of Sociology*, 98(6):1259–1319.

Papakostas, A. and Tollis, I. G. (1996). Issues in interactive orthogonal graph drawing. In Brandenburg (1996), pages 419–430.

Patrignani, M. (1999). On the complexity of orthogonal compaction. Technical Report RT-DIA-39-99, Dipartimento di Informatica e Automazione, Università degli Studi di Roma Tre. To appear in *Proceedings of WADS '99*.

Pelillo, M., editor (1997). *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR '97)*, volume 1223 of *Lecture Notes in Computer Science*. Springer.

Purchase, H. C. (1997). Which aesthetic has the greatest effect on human understanding? In Di Battista (1997), pages 248–261.

Purchase, H. C., Cohen, R. F., and James, M. (1996). Validating graph drawing aesthetics. In Brandenburg (1996), pages 435–446.

Purchase, H. C., Cohen, R. F., and James, M. (1997). An experimental study of the basis for graph drawing algorithms. *ACM Journal of Experimental Algorithmics*, 2(4).

Quinn, N. R. and Breuer, M. A. (1979). A force directed component placement procedure for printed circuit boards. *IEEE Transactions on Circuits and Systems*, 26(6):377–388.

Raab, J. (1999). *Steuerungsstrukturen politisierter Großprivatisierungen in Ostdeutschland. Das Beispiel der Werft- und Stahlindustrien*. PhD thesis, University of Konstanz. In preparation.

Richards, W. D. and Seary, A. J. (1997). Introduction to eigen analysis of networks. Paper presented at the *17th International Conference on Social Network Analysis (Sunbelt XVII)*, San Diego. Available at `http://www.sfu.ca/~richards/wdr97.htm`.

Roethlisberger, F. J. and Dickson, W. J. (1939). *Management and the Worker*. Harvard University Press.

Sabidussi, G. (1966). The centrality index of a graph. *Psychometrika*, 31:581–603.

Scarini, P. (1996). Elaborations of the Swiss agricultural policy for the GATT negotiations: A network analysis. *Swiss Journal of Sociology*, 22(1):85–115.

Schnyder, W. (1990). Embedding planar graphs on the grid. In *Proceedings of the 1st ACM-SIAM Symposium on Discrete Algorithms (SODA '90)*, pages 138–148.

Scott, J. (1991). *Social Network Analysis: A Handbook*. Sage Publications.

Sigl, R. (1969). *Ebene und sphärische Trigonometrie*. Akademische Verlagsgesellschaft.

Stein, S. (1951). Convex maps. *Proceedings of the American Mathematical Society*, 2:464–466.

Steinitz, E. and Rademacher, H. (1934). *Vorlesungen über die Theorie der Polyeder*. Springer.

Sugiyama, K. and Misue, K. (1995). A simple and unified method for drawing graphs: Magnetic-spring algorithm. In Tamassia and Tollis (1995), pages 364–375.

Sugiyama, K., Tagawa, S., and Toda, M. (1981). Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man and Cybernetics*, 11(2):109–125.

Tamassia, R. (1987). On embedding a graph in the grid with the minimum number of bends. *SIAM Journal on Computing*, 16:421–444.

Tamassia, R., Di Battista, G., and Batini, C. (1988). Automatic graph drawing and readability of diagrams. *IEEE Transactions on Systems, Man and Cybernetics*, 18(1):61–79.

Tamassia, R. and Tollis, I. G., editors (1995). *Proceedings of the 2nd International Symposium on Graph Drawing (GD '94)*, volume 894 of *Lecture Notes in Computer Science*. Springer.

Thomassen, C. (1980). Planarity and duality of finite and infinite planar graphs. *Journal of Combinatorial Theory, Series B*, 29:244–271.

Tilling, L. (1975). Early experimental graphs. *British Journal of Historical Science*, 8:193–213.

Trent, H. (1954). A note on the enumeration and listing of all possible trees in a connected linear graph. *Proceedings of the National Academy of Science, U.S.A.*, 58:72–76.

Tufte, E. R. (1983). *The Visual Display of Quantitative Information.* Graphics Press.

Tufte, E. R. (1990). *Envisioning Information.* Graphics Press.

Tufte, E. R. (1997). *Visual Explanations: Images and Quantities, Evidence and Narrative.* Graphics Press.

Tukey, J. W. (1977). *Exploratory Data Analysis.* Addison–Wesley.

Tunkelang, D. (1994). A practical approach to drawing undirected graphs. Technical Report CMU-CS-94-161, School of Computer Science, Carnegie Mellon University.

Tutte, W. T. (1963). How to draw a graph. *Proceedings of the London Mathematical Society, Third Series*, 13:743–768.

van Laarhoven, P. J. M. and Aarts, E. H. L. (1988). *Simulated Annealing: Theory and Applications.* Reidel.

Vijayan, G. (1986). Geometry of planar graphs with angles. In *Proceedings of the 2nd ACM Annual Symposium on Computational Geometry (SoCG '86)*, pages 116–124. ACM, The Association for Computing Machinery.

Wagner, K. (1936). Bemerkungen zum Vierfarbenproblem. *Jahresbericht der Deutschen Mathematikervereinigung*, 46:26–32.

Wainer, H. (1997). *Visual Revelations.* Copernicus.

Wasserman, S. and Faust, K. (1994). *Social Network Analysis: Methods and Applications.* Cambridge University Press.

Webber, R. (1997). Finding the best viewpoints for three-dimensional graph drawings. In Di Battista (1997), pages 87–98.

Webber, R. (1998). *Finding the Best Viewpoint for Three-Dimensional Graph Drawings.* PhD thesis, University of Newcastle. See `http://www.cs.mu.oz.au/~rwebber/research/thesis/`.

Whitesides, S. H., editor (1998). *Proceedings of the 6th International Symposium on Graph Drawing (GD '98)*, volume 1547 of *Lecture Notes in Computer Science.* Springer.

Winkler, G. (1995). *Image Analysis, Random Fields and Dynamic Monte Carlo Methods*, volume 27 of *Applications of Mathematics.* Springer.

Wood, D. (1996). On higher-dimensional orthogonal graph drawing. Technical Report 96/286, Department of Computer Science, Monash University. Also in *Proceedings of CATS '97, Computing: The Australasian Theory Symposium*, pages 3–8.

List of Symbols

Index