Journal of Graph Algorithms and Applications
http://jgaa.info/
vol. 7, no. 2, pp. 181–201 (2003)

Visual Ranking of Link Structures

Ulrik Brandes
Department of Mathematics & Computer Science
University of Passau
http://algo.fmi.uni-passau.de/~brandes/
brandes@algo.fmi.uni-passau.de

Sabine Cornelsen
Department of Computer & Information Science
University of Konstanz
http://www.inf.uni-konstanz.de/~cornelse/
cornelse@inf.uni-konstanz.de

Abstract

Methods for ranking World Wide Web resources according to their position in the link structure of the Web are receiving considerable attention, because they provide the first effective means for search engines to cope with the explosive growth and diversification of the Web. Closely related methods have been used in other disciplines for quite some time.

We propose a visualization method that supports the simultaneous exploration of a link structure and a ranking of its nodes by showing the result of the ranking algorithm in one dimension and using graph drawing techniques in the remaining one or two dimensions to show the underlying structure. We suggest to use a simple spectral layout algorithm, because it does not add to the complexity of an implementation already used for ranking, but nevertheless produces meaningful layouts. The effectiveness of our visualizations is demonstrated with example applications, in which they provide valuable insight into the link structure and the ranking mechanism alike. We consider them useful for the analysis of query results, maintenance of search engines, and evaluation of Web graph models.

Communicated by Giuseppe Liotta and Ioannis G. Tollis: submitted October 2001; revised December 2002.

Research supported in part by the Deutsche Forschungsgemeinschaft (DFG) under grant Br 2158/1-1 and the European Commission within FET Open Project COSIN (IST-2001-33555).

1 Introduction

The directed graph induced by the hyperlink structure of the Web has been recognized as a rich source of information. Understanding and exploiting this structure has a proven potential to help dealing with the explosive growth and diversification of the Web. Probably the most widely recognized example of this kind is the PageRank index employed by the Google search engine [9].

PageRank is but one of many models and algorithms to rank Web resources according to their position in a hyperlink structure (see, e.g., [36, 29, 13, 1, 8, 12]). We propose a method to complement rankings with a meaningful visualization of the graph they are computed on.

While graph visualization is an active area of research as well [14, 28], its integration with quantitative network analyses is only beginning to receive attention. It is, however, rather difficult to understand the determinants of, say, a particular ranking if its results do not influence the way in which the structure is visualized.

A design for graph visualizations showing a vertex valuation in its structural context is introduced in [6, 5]. In two-dimensional diagrams of social networks, the vertical dimension of the layout area is used to represent exactly the value assigned to each actor (a constraint), and a layout of the horizontal dimension is determined to make the diagram readable (an objective). Since the networks in question are relatively small (no more than a hundred vertices), an adaptation of the Sugiyama framework for layered graph drawing [38] is used for horizontal layout.

The guiding principle in the above design is axis separation: in one dimension the most important information is conveyed precisely, and in another the perception of its basis is eased. To facilitate visual exploration of ranking methods on larger link structures such as Web graphs, we propose to apply the same principle, but with a very different layout algorithm that is more appropriate for the specific type of data.

Standard rankings are based on spectral methods and iterative computation, but the same methods can also be used for graph layout. In the present application they are particularly well-suited, because densely connected subgraphs are clustered. On the Web, such subgraphs correspond to related resources and graphical clustering is therefore highly desirable. By using the axis separation principle and spectral layout techniques, a uniform approach to visual ranking of link structures is achieved.

The paper is organized as follows. In Section 2, we recall some fundamental spectral properties of graphs. Link-based ranking is surveyed in Section 3, and formally and computationally similar layout techniques are described in Section 4. Applications in which our visualization approach may be useful are discussed in Section 5 and examples with generated and real-world data are provided. We conclude in Section 6.

2 Preliminaries

The structural features of the Web can be captured in a directed graph $G = (V, E)$, where the set V of vertices represents the set of resources on the Web, and there is a directed edge $(u, v) \in E$ from a resource u to a resource v, if u contains a hyperlink to v. All graphs considered in this paper are assumed to be connected. We do not allow parallel edges, but self-loops and a positive real weight ω_{uv} for every edge. Let $A(G) = A = (A_{uv})_{u,v \in V}$ be the *adjacency matrix* of a graph, i.e. $A_{uv} = \omega_{uv}$ if $(u, v) \in E$, and $A_{uv} = 0$ otherwise. The *indegree* (*outdegree*), d_v^+ (d_v^-), of a vertex $v \in V$ is $\sum_{u:(u,v)\in E} A_{uv}$ ($\sum_{w:(v,w)\in E} A_{vw}$).

We will make extensive use of algebraic properties of graphs. If A is a square matrix and $Ap = \lambda p$, λ is called an *eigenvalue* of A and p an associated *eigenvector*. Note that, if p is an eigenvector associated with λ, then cp, $c \in \mathrm{R}$, is also. The *multiplicity* of an eigenvalue is the number of distinct eigenvectors associated with it. Counting multiplicities, an $n \times n$ matrix has n eigenvalues. The multiset $\Lambda(A) = \{\lambda_1, \ldots, \lambda_n\}$ of its eigenvalues with their respective multiplicity is called the *spectrum* of A.

We recall some important properties of spectra. The following lemma applies in particular to adjacency matrices of undirected graphs.

Lemma 1 *Let A be a real symmetric $n \times n$ matrix.*

1. *All eigenvalues of A are real.*
2. *Any two eigenvectors of A with distinct eigenvalues are orthogonal.*
3. *Let $\Lambda(A) = \{\lambda_1, \ldots, \lambda_n\}$, then*
   (a) *$\Lambda(cA) = \{c\lambda_1, \ldots, c\lambda_n\}$ for all $c \in \mathrm{R}$, in particular $\Lambda(-A) = \{-\lambda_1, \ldots, -\lambda_n\}$, and*
   (b) *$\Lambda(I + A) = \{1 + \lambda_1, \ldots, 1 + \lambda_n\}$.*

For directed graphs, we have the following theorem, which is a version of the fundamental Perron-Frobenius Theorem reformulated for our purposes.

Theorem 2 *If A is the adjacency matrix of a strongly connected graph G, then there is an ordering $\lambda_1 \geq |\lambda_2| \geq \cdots \geq |\lambda_n|$ of its eigenvalues such that λ_1 is real and simple, and $-\lambda_1$ is an eigenvalue of A if and only if G is bipartite. Moreover, the entries of a non-zero eigenvector associated with λ_1 are either all negative or all positive real numbers.*

For further background on matrix computations and algebraic properties of graphs we refer to [21] and [20].

3 Structural Ranking of Web Resources

Any real-valued vector $p = (p_v)_{v \in V}$ defined on the vertices of a graph is called a *prominence index*, where p_v is the *prominence* of vertex v. A *ranking* is obtained

from a prominence index by ordering the vertices according to non-increasing prominence.

Many models have been proposed to capture an explicitly or implicitly defined notion of a vertex's prominence in a graph [27, 25, 16, 4, 17, 36, 29, 1, 13, 12, and many more]. Though in general only defined for undirected graphs, we first outline eigenvector centrality, because it nicely illustrates some important commonalities of the popular ranking methods that we discuss below.

Assume that the prominence of a vertex is understood to be proportional to the combined prominence of its neighbors, $\lambda p_v = \sum_{u:\{u,v\}\in E} \omega_{uv} p_u$, $v \in V$, where the constant λ is introduced so that the system of equations has a non-zero solution. This definition yields the eigensystem of the (transposed) adjacency matrix,

$$\lambda p = A^T p = Ap, \qquad \text{(\textit{eigenvector centrality} [3])}$$

and every eigenvector of $A = A(G)$ gives a ranking of the vertices for the above notion of prominence, although the *principal eigenvector*, i.e. the one associated with the eigenvalue of largest magnitude, is generally preferred [4, 17]. The principal eigenvector can be obtained by power iteration, which starts with any non-zero vector and iteratively multiplies the matrix with the current solution, e.g. $p^{(0)} \leftarrow \mathbf{1}$ and

$$p^{(k+1)} \quad \leftarrow \quad A \cdot p^{(k)}.$$

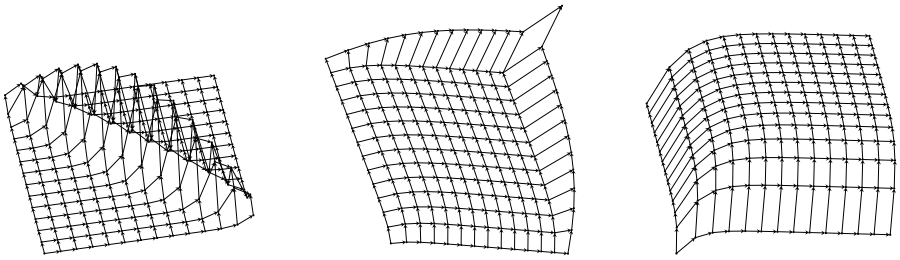
Since the matrices considered here originate from large and sparse graphs, multiplication is carried out by computing $p_v^{(k+1)} \leftarrow \sum_{u:\{u,v\}\in E} \omega_{uv} p_u^{(k)}$ for every $v \in V$. To prevent the entries of the iterates from growing out of range, each vector is normalized such that the magnitude of the largest entry equals the number of vertices in the graph (recall that multiples of eigenvectors are eigenvectors as well). This normalization scheme is applied in all subsequently described iterative computations without explicit mentioning.

More elaborate indices defined on directed graphs are discussed below. In Figure 1 they are illustrated on an acyclic grid. The grid is placed in a plane and each grid point is then lifted according to its prominence.

3.1 Hubs and authorities

A natural notion of prominence for a Web resource is the extent to which it is referred to by other Web pages, in particular by those pages that specialize in listing useful resources. In turn, the property of being such a list of useful resources is a notion of prominence in itself. In these complementary and mutually reinforcing notions prominent resources are called *authorities* (resources with useful information) and *hubs* (pages with useful links).

The hub score of a page is proportional to the combined authority of the resources it links to, and the authority of a resource is proportional to the combined hub score of the pages linking to it. In practice, hub and authority



(a) authority (b) PageRank (c) Katz's status

Figure 1: Comparison of prominence indices on a directed grid

scores are thus computed by iterating $p^{(0)} \leftarrow \mathbf{1}$ and

$$\begin{aligned} p^{(2k+1)} &\leftarrow A^T \cdot p^{(2k)} \\ p^{(2k+2)} &\leftarrow A \cdot p^{(2k+1)}. \end{aligned}$$

For $h^{(k)} = p^{(2k)}$ and $a^{(k)} = p^{(2k+1)}$, the alternating iteration can be written as

$$h^{(k+1)} \leftarrow AA^T \cdot h^{(k)} \qquad (\textit{hubs}\ [29])$$

$$a^{(k+1)} \leftarrow A^TA \cdot a^{(k)}. \qquad (\textit{authorities}\ [29])$$

In this formulation, it is easy to see that the hub and authority indices in a graph with adjacency matrix A correspond to eigenvector centrality in the weighted undirected graphs with symmetric adjacency matrix AA^T and A^TA, respectively.

As can be seen in Figure 1(a), vertices on and above the falling diagonal of the grid have the highest authority, because they are in the midst of the undirected graph induced by A^TA. Compare this to the undirected graph induced by AA^T, indicating why the best hubs are found on and below this diagonal.

3.2 PageRank

In another variant of eigenvector centrality the contribution of each vertex to another vertex's prominence is weighted by its outdegree, $p_v = \sum_{u:(u,v)\in E} \frac{\omega_{uv} p_u}{d_u^+}$ (see e.g. [35, 11]). If we require p to be a probability distribution over the set of vertices, this notion has a nice interpretation as the stationary distribution of the simple random walk on the graph (or random surfer on the Web, if you will), in which each edge leaving a vertex is chosen with equal probability.

Let $M = D^{-1}A$ be the adjacency matrix normalized so that the rows sum to one, where D is the diagonal matrix with the outdegrees on the diagonal. Then, M is a stochastic matrix of transition probabilities, and a stationary

distribution $p = M^T \cdot p$ satisfies the above notion of prominence. However, if a vertex has outdegree zero, the computation breaks down, and strongly connected components may cause an overdue increase of the prominence of their vertices. This so-called "sink problem" can be avoided by introducing an escape mechanism. Let $\hat{p}$ be an a-priori probability distribution over the vertices (e.g., user preferences or general popularity of a resource), then with probability ω the random walk picks an edge of the graph whereas with the remaining probability, it jumps to any other vertex according to $\hat{p}$. The index is thus defined by

$$\begin{aligned} p &= \omega M^T p + (1-\omega)\hat{p} && (\textit{PageRank}\ [8]) \\ &= \left(\omega M^T + (1-\omega)\hat{p} \cdot \mathbf{1}^T\right) \cdot p. \end{aligned}$$

The second equality holds because p is a probability distribution. From the second expression it can be seen that PageRank is the eigenvector centrality of a weighted graph with a complete set of additional escape edges. This modified matrix is irreducible and aperiodic so that the iteration $p^{(0)} \leftarrow \frac{1}{n}\mathbf{1}$ and

$$p^{(k+1)} \leftarrow \left(\omega M + (1-\omega)\mathbf{1} \cdot \hat{p}^T\right)^T \cdot p^{(k)}$$

converges to a unique prominence vector. On the grid in Figure 1(b), the random surfer may jump to any vertex, but is most likely to walk towards the upper and right side of the grid, from where the only continuation is towards the upper right corner.

3.3 Katz's status index

As a generalization of simply using indegrees to measure 'status' in social networks, the prominence of a vertex is determined by the number of directed paths of arbitrary length ending in the vertex, where the influence of longer paths is attenuated by a decay factor. Recall that the entries of the k-th power of the adjacency matrix of an unweighted graph give the number of paths of length k between every pair of vertices. Therefore, this notion of prominence is determined by

$$p = \left(\sum_{k=1}^{\infty} (\alpha A^T)^k\right) \cdot \mathbf{1}, \qquad (\textit{Katz's status}\ [27])$$

where parameter α corresponds to the fraction of status that is passed along a directed edge. For sufficiently small values of α (a convenient choice is $\frac{1}{\Delta+1}$, where Δ is the minimum of the maximum in- or outdegree of any vertex in the graph), the sum converges to $(I - \alpha A^T)^{-1} - I$. Therefore, the status vector can be obtained by solving $\left(\alpha^{-1} I - A^T\right) \cdot p = d$, where d is the vector of indegrees. Solving this system of linear equations directly is prohibitive for large graphs. Standard sparse matrix approaches approximate a solution iteratively. The update step in Jacobi iteration, for instance, yields $p^{(k+1)} \leftarrow \alpha A^T \cdot p^{(k)} + \alpha d$. This iteration nicely reflects the underlying notion of adding contributions from

vertices farther and farther away. The same can be observed in Figure 1(c), where the attenuated influence from vertices in the lower left does not suffice to discriminate the prominence of vertices in the upper right any more.

In a sense, the above definitions of prominence are contained in the following generic formulation of status in networks. It puts a twist on eigenvector centrality through the addition of an a-priori prominence vector $\hat{p}$,

$$p = A^T p + \hat{p}. \qquad (\textit{Hubbel's status } [25])$$

By choosing appropriate weights and a-priori prominences, we obtain eigenvector centrality and PageRank. Reordering, we have $p = (I - A^T)^{-1} \cdot \hat{p}$, provided the inverse exists. If it does, it equals $\sum_{k=0}^{\infty}(A^T)^k$, and therefore $p = \left(\sum_{k=0}^{\infty}(A^T)^k\right) \cdot \hat{p} = \left(I + \sum_{k=1}^{\infty}(A^T)^k\right) \cdot \hat{p}$. With uniform edge weights and $\hat{p} = \mathbf{1}$ we obtain a prominence index in which every component is by one larger than Katz's status index.

4 Spectral Graph Layout

In the previous section we emphasized formal similarities in the definition of popular prominence indices. In practice, all of them are computed by some variant of sparse matrix power iteration, i.e. by iterating over all vertices, and, for each vertex, combining the current scores of its neighbors. Implementation of these algorithms is thus trivial.

In this section, we introduce a layout algorithm that produces meaningful layouts using the same principles as the ranking methods. It is therefore a simple matter to complement an existing system for ranking vertices to compute a layout of the graph on the fly, since both parts of the system can operate synchronously on the same data

4.1 Layout with eigenvectors

For layout, we consider the unweighted, undirected, simple graph obtained by omitting weights, directions, self-loops, and multiple edges. Note that edge directions are sufficiently represented in the prominence dimension.

Let A be the adjacency matrix of a simple undirected graph G and $D = D(G)$ its diagonal *degree matrix*. We consider the *Laplacian matrix* $L = D - A$, which has interesting applications in many areas (see, e.g., [33]). Its usefulness for drawing graphs was first described in [22] and is based on the observation that minimizing the associated quadratic form

$$x^T L x = \sum_{\{u,v\} \in E} (x_u - x_v)^2, \qquad (1)$$

corresponds to minimizing the squared distance between pairs of adjacent vertices if x is interpreted as a vector of vertex positions. This objective functions

is closely related to standard graph drawing methods, since it can be interpreted as the energy of a physical system consisting of rings (the vertices) that are tied together by springs (the edges) of natural length zero. In other words, we have a spring-embedder with zero-length springs and no repelling forces.

The energy-minimum state of the above system is obtained by assigning the same position to all vertices (recall that we assume connectedness of the graph). These undesirable single-point solutions can be avoided by fixing some selected vertices at distinct positions. Minimization subject to these boundary conditions yields the well-known barycentric layout model of Tutte [39]. However, the final layout is contingent on the fixed vertices and their position. While placing a face of a triconnected planar graph on a convex polygon yields a planar layout of the graph, there are no general rules on which vertices to place where in more general graphs.

Other alternatives include the addition of repulsive forces between nodes [15, 18] and the use of springs with non-zero length [26]. Although these methods have been extended to be applicable on graphs with thousands of vertices [19, 23, 40], there implementation is far from trivial.

Spectral methods take a different approach and yield a trivial, parameter-free algorithm working toward a globally optimal solution with respect to the above quadratic objective function. Note that the undesired minima $x = c\mathbf{1}$ are the eigenvectors associated with eigenvalue zero, i.e. $Lx = 0$. More generally, if (λ, x) is any eigenpair of L, then $\lambda = \frac{x^T L x}{x^T x}$. We therefore want to minimize

$$\frac{x^T L x}{x^T x} \text{ subject to } \mathbf{0} \neq x \perp \mathbf{1},$$

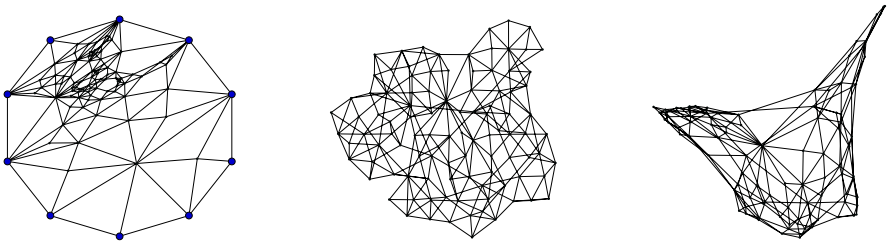
since the eigenvectors of a symmetric matrix are orthogonal. Hence, the desired solution is an eigenvector associated with the second-smallest eigenvalue of L. This vector is called the *Fiedler vector* and, because of its distance minimization property, frequently used in graph partitioning (see, e.g., [37]). For the same reason, it yields a useful one-dimensional layout of a graph, because edges are short and hence dense subgraphs are clustered. Another argument in favor of using the Fiedler vector for horizontal layout is its successful application in drawing bipartite graphs in two-layers with few crossing edges [34].

If rankings ought to be visualized in three dimensions (cf. Figure 1), a reasonable choice for the second free dimension is the normalized eigenvector minimizing the objective function subject to being orthogonal to $\mathbf{1}$ and the first solution.

An example of two-dimensional layouts obtained from barycentric layout, a typical spring embedder, and two orthogonal eigenvectors of L is provided in Figure 2. While the spring embedder produces more uniform edge lengths, the eigenvectors emphasize structural clustering of vertices.

4.2 Computing the layout

Eigenvectors associated with the smallest eigenvalues of large sparse matrices are usually computed using Lanczos' method. However, all popular prominence



(a) barycentric (b) spring embedder (c) spectral

Figure 2: Two-dimensional layouts of a random planar triconnected graph

indices are computed using a variant of the much simpler power iteration, which only gives an eigenvector associated with the eigenvalue of largest magnitude. To be able to apply the same simple algorithm and thus synchronize layout and prominence computation, we reverse the eigenvalues of the Laplacian.

By Lemma 1, all eigenvalues of L are real, and since L is positive-semidefinite they are non-negative. By Gershgorin's Theorem, the largest eigenvalue is no more than twice the maximum vertex degree Δ of the graph, so that again by Lemma 1 the matrix $L' = 2\Delta \cdot I - L$ has the same eigenvectors as L, but the order of the corresponding eigenvalues is reversed.

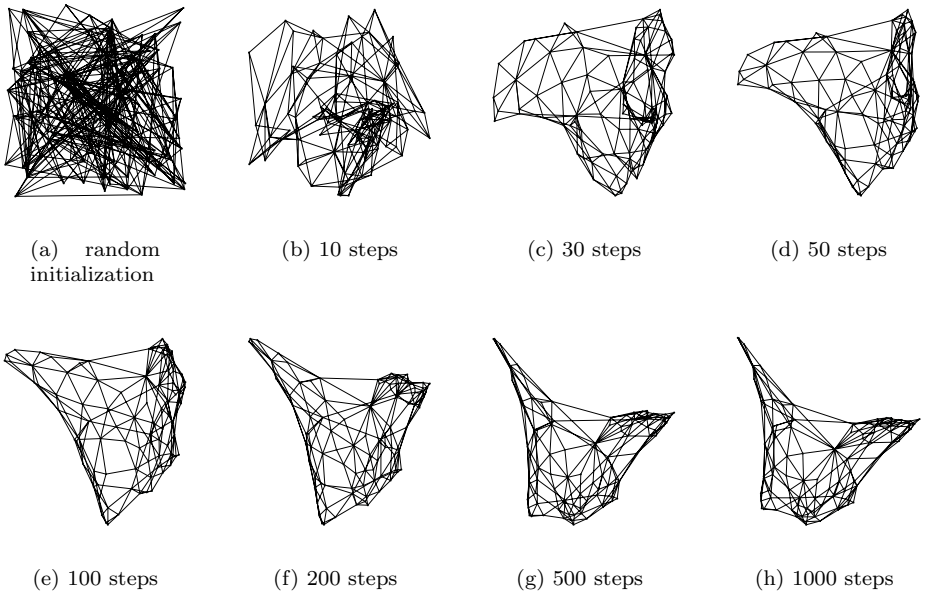
Straightforward application of power iteration on L' returns the principal eigenvector of L', which is the trivial eigenvector $\mathbf{1}$ associated with the smallest eigenvalue of L. Power iteration on a vector that is orthogonal to the principal eigenvector yields an eigenvector of the second-largest eigenvalue of L', and hence the desired layout for the first dimension. If needed, iterating on a vector that is orthogonal to both the trivial eigenvector and the approximate solution for the first dimension yields the second dimension.

A vector y is orthogonalized with respect to another vector x by setting $y \leftarrow y - \frac{x^T \cdot y}{x^T \cdot x} x$. Orthogonalization with respect to the trivial eigenvector $\mathbf{1}$ is even easier, since it corresponds to subtracting, from each entry of y, the mean of all its entries. To obtain vectors x and y for a two-dimensional layout we thus carry out the following augmented power iteration on random starting vectors $x^{(0)}, y^{(0)}$ that are repeatedly orthogonalized with respect to $\mathbf{1}$ and to one another

$$x^{(k+1)} \leftarrow L' \cdot x^{(k)}; \qquad x^{(k+1)} \leftarrow x^{(k+1)} - \frac{1}{n} \sum_{v \in V} x_v^{(k+1)}$$

$$y^{(k+1)} \leftarrow L' \cdot y^{(k)}; \qquad y^{(k+1)} \leftarrow y^{(k+1)} - \frac{1}{n} \sum_{v \in V} y_v^{(k+1)}$$

$$y^{(k+1)} \leftarrow y^{(k+1)} - \frac{x^{(k+1)T} \cdot y^{(k+1)}}{x^{(k+1)T} \cdot x^{(k+1)}} x^{(k+1)}$$



(a) random initialization (b) 10 steps (c) 30 steps (d) 50 steps

(e) 100 steps (f) 200 steps (g) 500 steps (h) 1000 steps

Figure 3: Typical convergence behavior of the power iteration for layout. Note that x-coordinates do not change significantly after 30 iterations

Intuitively, the layout is centered, rectified, and (due to the normalization) zoomed after each multiplication with L'. The last two lines are omitted if only one dimension needs to be determined for the layout.

Note that in our setting the potentially slow convergence of power iteration is of minor importance since all we are looking for is a vector that approximately minimizes the quadratic objective function (1). Though overall convergence depends on the ratio of the largest eigenvalues, the iterate quickly moves toward a subspace spanned by eigenvectors associated with large eigenvalues. Only then, when the largest eigenvalues need to be separated, does the slow-down take effect. Figure 3 gives a typical, qualitative example.

As a quantitative measure of convergence we use the *residual* $r(x) = \|L'x - \frac{x^T L' x}{x^T x} x\|^2$, that is the squared distance of x from being an eigenvector associated with the current eigenvalue estimate $\frac{x^T L' x}{x^T x}$. Recall that we normalize after each iteration such that the magnitude of the largest entry (the largest coordinate) equals the number of vertices. We consider a layout to be of sufficient quality, if the residual is of the same order, i.e. if on the average each vertex is one unit off its optimal position. The entire one-dimensional layout algorithm is given in Algorithm 1. Note that it requires no external parameters, and is trivial to implement along with a ranking algorithm.

We compared the number of iterations needed for layout to that needed for ranking. Since ranking is the important information to be conveyed, it is

Algorithm 1: One-dimensional spectral layout

Input: simple, connected, undirected graph $G = (V, E)$, $n = |V|$

Output: one-dimensional layout $x = (x_v)_{v \in V}$

$r = \infty$;
$x \leftarrow n \cdot \mathbf{1}$;
while $\frac{r}{n} > 1$ **do**
  $x' \leftarrow L'x$;
  $x' \leftarrow x' - \frac{\sum_{v \in V} x'_v}{n} \cdot \mathbf{1}$;
  $r \leftarrow \|x' - \frac{x^T x'}{x^T x} x\|^2$;
  $x \leftarrow \frac{n}{\max_{v \in V} x'_v} \cdot x'$;

required to be precise. Convergence of ranking iterations is assumed when the corresponding residual is below 1 (rather than the number of vertices). Our experience suggests that the number of iterations needed for the layout is larger than that for ranking, but not by much. In Figure 4, convergence of ranking and layout is compared on example graphs.

For larger graphs with tens of thousands of nodes, the simple algorithm nevertheless becomes to slow, especially when compared with the fastest-converging ranking algorithms. A much more sophisticated multiscale algorithm [30] to obtain the Fiedler vector is available, though.

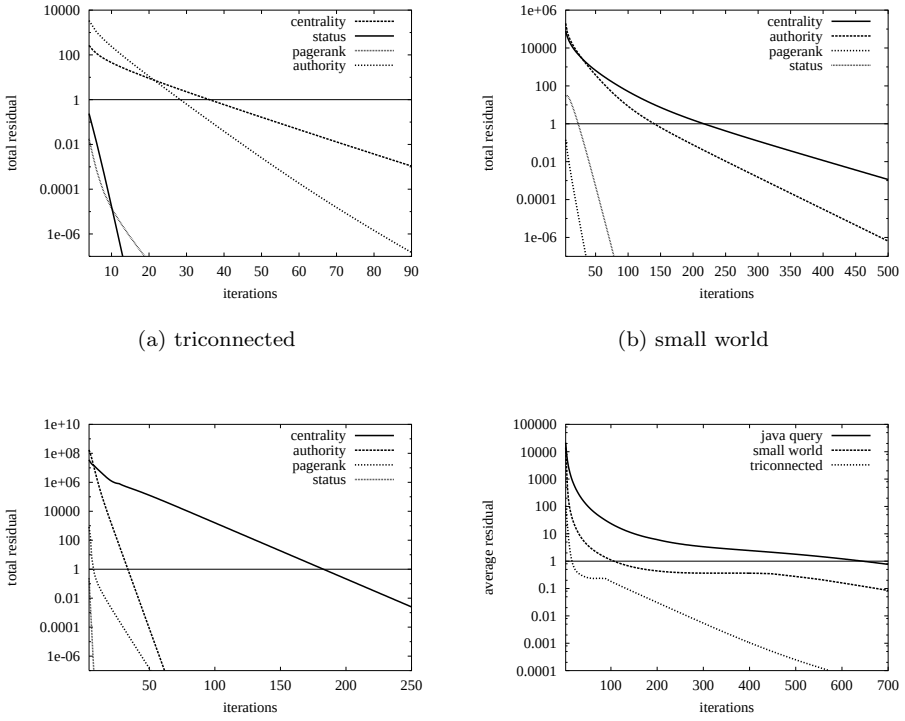
5 Application Examples

We demonstrate our visualization approach on three different kinds of data: random Web graphs generated from popular models, a search engine example constructed from an AltaVista query, and a bibliographic data set. Our C++-implementations use LEDA, the *Library of Efficient Data Types and Algorithms* [32].

5.1 Web graph models

In the *linear growth model* [31], a graph grows one vertex at a time. At each time step, a prototype is chosen among already existing vertices, and a new vertex is generated. This new vertex is then assigned a fixed number of outgoing edges. With some fixed probability, the ith of these edges points to a randomly selected vertex among those already existing (creation case), and with the remaining probability it points to the same vertex as the ith outgoing edge of the prototype vertex (copying case). Our generator does not introduce multiple edges, and if a prototype happens to not have enough outgoing edges, no edge is introduced in the copying case. Clearly, all graphs evolving like this are acyclic.

In the *exponential growth model* [31], a graph grows by a fixed fraction of its current size at each time step. New vertices receive a fixed number of loops,



(a) triconnected

(b) small world

(c) "java" query

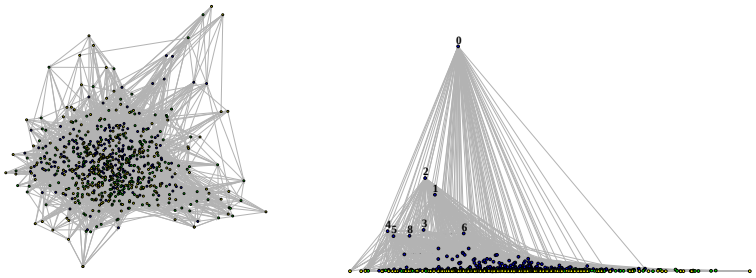
(d) spectral layout

Figure 4: Convergence of ranking and layout compared (logarithmic scale). The three graphs are the triconnected planar graph of Figure 2, the random small world of Figure 5(c) (Section 5.1), and the Web graph of Figure 6 (Section 5.2)

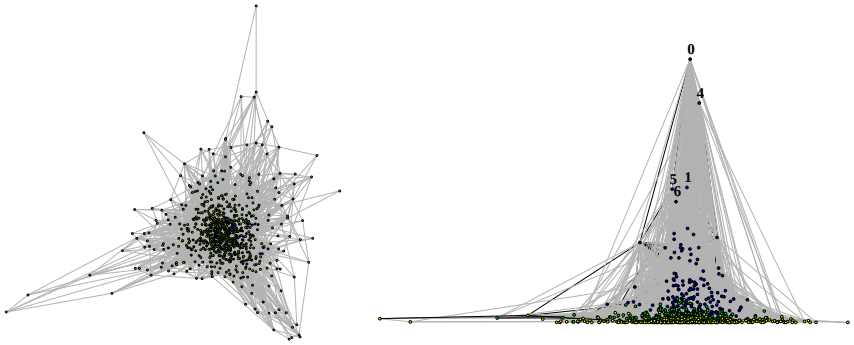
and for each already existing edge, its target receives a new incoming edge for which, with some fixed probability, the source is chosen uniformly at random from the new vertices, and otherwise from the existing vertices with probability proportional to their current outdegree. We used a simpler model in which existing vertices are chosen uniformly at random as well.

For the *small-world model* [41], we initially generate a cyclic sequence of vertices and let a vertex link to a fixed number of predecessors and successors. Then, each edge is rewired with some small probability by choosing a new destination uniformly at random.

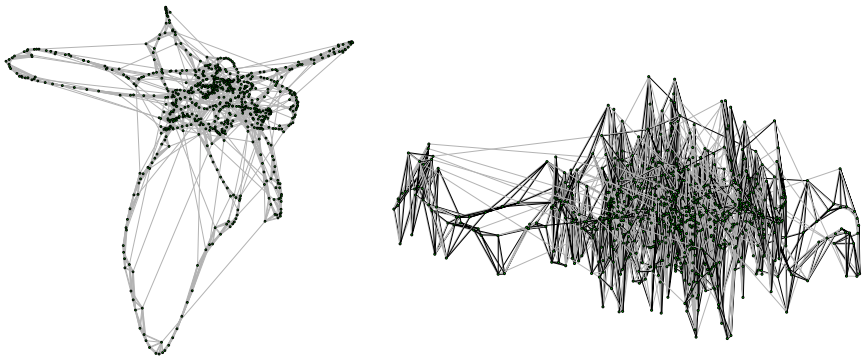
Figure 5 shows spectral layouts of graphs generated according to these models and rankings replacing the vertical dimension with PageRank as an example of a prominence index. The linear model graph has about 750 vertices and was generated with desired outdegree 7 and copying probability 0.3, where some of the vertices created last where trimmed because of poor connectivity. The expo-



(a) Linear growth evolving copying model [31]



(b) Exponential growth evolving copying model [31]



(c) Small-world model [41]

Figure 5: Web models (2D spectral layout and 1D spectral layout vs. PageRank)

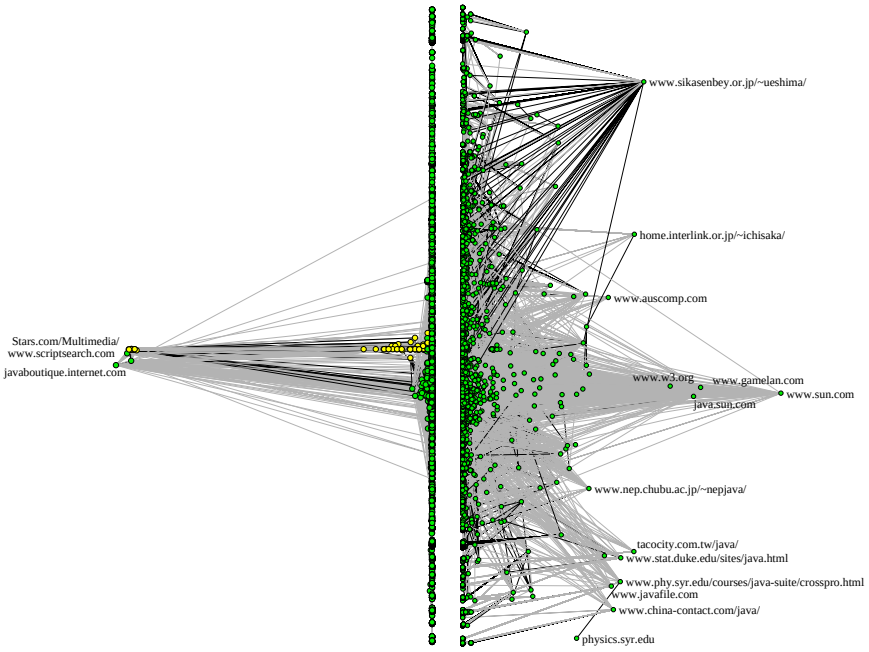


Figure 6: Authority and PageRank visualization of "java" query result

nential model graph was generated with 10% growth rate, desired outdegree 7, 7 initial loops, and probability 0.5 for choosing the origin among new vertices. It originally had about 1000 vertices, but again roughly a quarter of the vertices last created were removed to achieve more robust structure. The 750 vertices in the small-world graph originally linked to their six nearest cyclic neighbors and edges were rewired with probability 0.05. In all rankings, edge directions are indicated by color (gray edges point upwards, black edges point downwards).

There is no visible clustering in the evolving copying models. Moreover, the prominence of resources appears to be correlated with their age (also with the other indices outlined in Section 3). The figures thus graphically support the conclusion of [31] that *death processes*, i.e. the occasional deletion of vertices and edges, might be necessary for the evolving copying models to be realistic. In the small-world model, the spectral layout reveals a cycle crumpled by chords, and the ranking shows that the model yields a rather egalitarian structure.

Our generators are slightly simplified versions of the original ones and our samples are not representative. Their sole purpose is to demonstrate the potential utility of ranked visualizations in the exploration and comparison of different models and parameterizations.

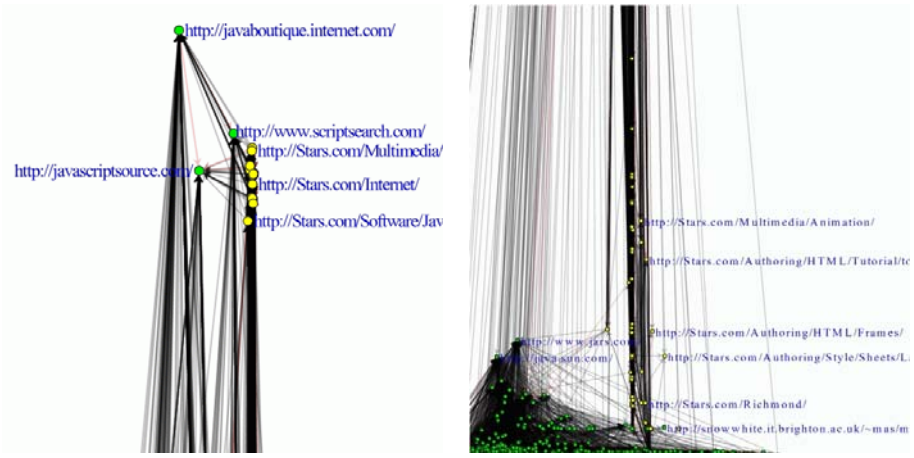


Figure 7: Detecting a data preparation flaw by visual inspection

5.2 Search engine query results

The data for this example was compiled in a way similar to the HITS algorithm [29]. We asked the AltaVista search engine for pages containing the word "java" and used the first 200 URLs it returned as the root set. It was then expanded by asking AltaVista for pages containing links to resources in the root set (backward extension), and adding resources linked to by pages in the root set (forward extension). The graph was completed by adding edges for all links between pages in the resulting set of vertices. The computations were carried out on the only large component of this graph from which some poorly connected vertices were removed to prevent extreme clustering. The graph has more than 5000 vertices and 15000 edges.

In Figure 6, this graph is shown twice, with vertices positioned vertically according to the Fiedler vector, and horizontally according to one of two prominence indices. Again, links from more to less prominent resources are colored black.

The most prominent resources under the PageRank measure match our expectations, but there are some surprising recommendations as well. It is clearly visible that some of these serve distinct user groups, like the Japanese directory in the upper right. Note that, without zooming into the image, we may not conclude that vertically close vertices are closely connected. However, it is safe to assume that vertically separated vertices are relatively distant in the structure. This feature can serve to distinguish query results which contain a keyword that is used in different contexts (see the "jaguar"-query example in [29]).

Figure 6 also shows that the top authorities are surprisingly distinguished from the rest of the graph, and quite different from our expectations. Most of them are located at `Stars.com`, a large repository for developers ("Web Developer's Virtual Library"). Since they are well connected among each other,

it is by virtue of our layout approach that their vertical position is similar, and thus this phenomenon could be detected by visual exploration. In Figure 7, resources at this site are colored lighter. Not surprisingly, vertices with high hub scores are from this site as well. This simple example graphically explains why the original HITS algorithm does not consider links within a site.

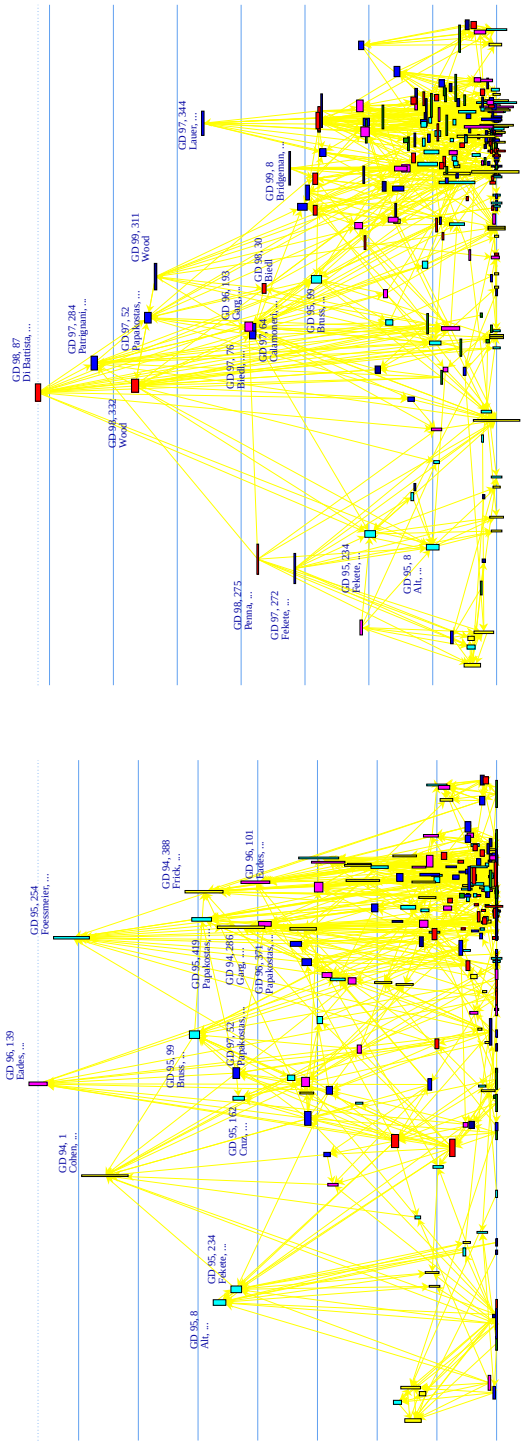
5.3 Bibliographic networks

Web graphs may be viewed as citation networks, and there exist many other bibliographic relations between publications and authors of written works. A discussion of techniques to analyze bibliographic networks is beyond the scope of this paper, but there is evidence that carefully applied network analytic methods can provide insight into the structure of a research area by identifying, e.g., prominent works or thematic clusters. We refer the reader to [42] for an introduction to bibliographic analysis and to [10] for an example of a system constructing and visualizing graphs from various bibliographic relations.

The application of our visual ranking approach to bibliographic networks is illustrated by citation data made available for the 2001 Graph Drawing Contest [2], held in conjunction with the 9th International Symposium on Graph Drawing. Since bibliographic networks typically contain loosely connected subgraphs which are difficult for spectral approaches to draw properly, it is proposed in [7] to weaken the diagonal of the Laplacian matrix. This modification serves to spread vertices more uniformly.

Each vertex in Figure 8 represents a paper that appears in one of the proceedings of the symposia from 1994 and 2000. While the color indicates the year of the symposium, height and width represent the number of citations received and made, respectively. As noted in Section 3, Kleinberg's hub and authority scores correspond to eigenvector centrality scores in the undirected graphs AA^T and A^TA. In bibliometrics, these are known as the bibliographic coupling and co-citation graphs. A hub is thus a potential survey, while an authority is an influential paper. Note, however, that the specific data at hand is certainly not sufficient to draw valid conclusions about the role and importance of a publication.

We have chosen this data set because the emerging patterns even for this small data set happen to resemble at least some of our intuition about the field. In particular, the horizontal clustering produced by the spectral layout algorithm does indeed correspond to a thematic clustering. The small cluster in the far right, for instance, are the Graph Drawing Contest Reports, connected only to the mainstream papers that form the adjacent dense cluster. Moving to the left, topics change via orthogonal drawing and 3D to the less intensely studied visibility representations and proximity drawings.



(a) authorities (b) hubs (surveys)

Figure 8: Citations between papers in proceedings of symposia on Graph Drawing (data from the 2001 GD Contest [2])

6 Conclusions

We have proposed a method for Web graph visualization that provides unambiguous identification of prominent resources while showing the entire graph and its clustering. In the simplest approach, the layout for our visualizations can be computed synchronously with common link-based rankings.

We expect the proposed visualization design to be particularly useful for visual exploration of ranked structures, for teaching and experimenting with ranking procedures, and for evaluation and illustration of stochastic models of the Web graph.

For graphs with tens of thousands of vertices, power iteration becomes costly because of its slow convergence. While speed-up techniques that reorganize storage to reduce external memory access [24] carry over to the layout algorithm, more sophisticated layout algorithms are available. Several recently introduced methods [19, 40, 23] produce layouts similar to the spectral approach. With a new multiscale technique for eigenvector layout computation [30], however, our approach extends directly.

The main advantage of spectral graph layout, its correspondence with distance minimization and hence with clustering, becomes a drawback in cases where the underlying undirected graph is poorly connected, since denser subgraphs will be clustered in a very small interval. Experiments with modifications of the Laplacian matrix [7] suggest that this problem can be addressed without changing the iteration significantly.

Acknowledgments. We thank Marco Gaertler for collecting the "java"-query data used in Section 5.2.

References

[1] Krishna Bharat and Monika R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. In *Proc. 21st Ann. Intl. ACM SIGIR Conf. Research and Development in Information Retrieval*, pages 104–111, 1998.

[2] Therese C. Biedl and Franz J. Brandenburg. Graph-drawing contest report. In *Proc. 9th Intl. Symp. Graph Drawing (GD '01)*, Springer LNCS 2265, pages 513–522, 2002.

[3] Phillip Bonacich. Factoring and weighting approaches to status scores and clique identification. *Journal of Mathematical Sociology*, 2:113–120, 1972.

[4] Phillip Bonacich. Power and centrality: A family of measures. *American Journal of Sociology*, 92:1170–1182, 1987.

[5] Ulrik Brandes, Jörg Raab, and Dorothea Wagner. Exploratory network visualization: Simultaneous display of actor status and connections. *Journal of Social Structure*, 2(4), 2001.

[6] Ulrik Brandes and Dorothea Wagner. Contextual visualization of actor status in social networks. In *Data Visualization 2000. Proc. 2nd Joint Eurographics/IEEE TCVG Symp. Visualization (VisSym '00)*, pages 13–22. Springer, 2000.

[7] Ulrik Brandes and Thomas Willhalm. Visualization of bibliographic networks with a reshaped landscape metaphor. In *Proc. 4th Joint Eurographics/IEEE TCVG Symp. Visualization (VisSym '02)*, pages 159–164. ACM Press, 2002.

[8] Sergey Brin, Rajeev Motwani, Lawrence Page, and Terry Winograd. What can you do with a web in your pocket? *IEEE Bulletin of the Technical Committee on Data Engineering*, 21(2):37–47, 1998.

[9] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[10] Anne Büggemann-Klein, Rolf Klein, and Britta Landgraf. BibRelEx: Exploring bibiliographic databases by visualization of annotated content-based relations. *D-Lib Magazine*, 5(11), 1999.

[11] Ronald S. Burt. *Toward a Structural Theory of Action: Network Models of Social Structure, Perception, and Action*. Academic Press, 1982.

[12] Soumen Chakrabarti, Byron E. Dom, Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, Andrew S. Tomkins, David Gibson, and Jon M. Kleinberg. Mining the Web's link structure. *IEEE Computer*, 32(8):60–67, 1999.

[13] Soumen Chakrabarti, Byron E. Dom, Prabhakar Raghavan, Sridhar Rajagopalan, David Gibson, and Jon M. Kleinberg. Automatic resource compilation by analyzing hyperlink structure and associated text. *Computer Networks and ISDN Systems*, 30(1–7):65–74, 1998.

[14] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1999.

[15] Peter Eades. A heuristic for graph drawing. *Congressus Numerantium*, 42:149–160, 1984.

[16] Linton C. Freeman. Centrality in social networks: Conceptual clarification. *Social Networks*, 1:215–239, 1979.

[17] Noah E. Friedkin. Theoretical foundations for centrality measures. *American Journal of Sociology*, 96(6):1478–1504, May 1991.

[18] Thomas M.J. Fruchterman and Edward M. Reingold. Graph-drawing by force-directed placement. *Software—Practice and Experience*, 21(11):1129–1164, 1991.

[19] Pawel Gajer, Michael T. Goodrich, and Stephen G. Kobourov. A fast multi-dimensional algorithm for drawing large graphs. In *Proc. 8th Intl. Symp. Graph Drawing (GD 2000)*, Springer LNCS 1984, pages 211–221, 2001.

[20] Chris Godsil and Gordon Royle. *Algebraic Graph Theory*, volume 207 of *Graduate Texts in Mathematics*. Springer, 2001.

[21] Gene H. Golub and Charles F. van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

[22] Kenneth M. Hall. An r-dimensional quadratic placement algorithm. *Management Science*, 17(3):219–229, 1970.

[23] David Harel and Yehuda Koren. A fast multi-scale method for drawing large graphs. In *Proc. 8th Intl. Symp. Graph Drawing (GD 2000)*, Springer LNCS 1984, pages 183–196, 2001.

[24] Taher H. Haveliwala. Efficient computation of pagerank. Technical Report 1999-31, Database Group, Stanford University, 1999.

[25] Charles H. Hubbell. An input-output approach to clique identification. *Sociometry*, 28:377–399, 1965.

[26] Tomihisa Kamada and Satoru Kawai. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31:7–15, 1989.

[27] Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18:39–43, 1953.

[28] Michael Kaufmann and Dorothea Wagner, editors. *Drawing Graphs: Methods and Models*, volume 2025 of *Lecture Notes in Computer Science*. Springer, 2001.

[29] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the Association for Computing Machinery*, 46(5):604–632, September 1999.

[30] Yehuda Koren, Liran Carmel, and David Harel. ACE: A fast multiscale eigenvectors computation for drawing huge graphs. In *Proc. IEEE Symp. Information Visualization 2002 (InfoVis '02)*, pages 137–144, 2002.

[31] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D. Sivakumar, Andrew S. Tomkins, and Eli Upfal. Stochastic models for the Web graph. In *Proc. 41st Ann. IEEE Symp. Foundations of Computer Science (FOCS 2000)*, pages 57–65, 2000.

[32] Kurt Mehlhorn and Stefan Näher. *The LEDA Platform of Combinatorial and Geometric Computing*. Cambridge University Press, 1999.

[33] Bojan Mohar. Some applications of Laplace eigenvalues of graphs. In Gena Hahn and Gert Sabidussi, editors, *Graph Symmetry: Algebraic Methods and Applications*, NATO ASI Series C 497, pages 225–275. Kluwer, 1997.

[34] Matthew Newton, Ondrej Sýkora, and Imrich Vrťo. Two new heuristics for two-sided bipartite graph drawings. In Michael T. Goodrich and Stephen G. Kobourov, editors, *Proceedings of the 10th International Symposium on Graph Drawing (GD '02)*, volume 2528 of *Lecture Notes in Computer Science*. Springer, 2002.

[35] Gabriel Pinski and Francis Narin. Citation influence for journal aggregates of scientific publications: Theory, with applications to the literature of physics. *Information Processing and Management*, 12(5):297–312, 1976.

[36] Peter Pirolli, James Pitkow, and Ramana Rao. Silk from a sow's ear: Extracting usable structures from the Web. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI '96)*, pages 118–125, 1996.

[37] Daniel A. Spielman and Shang-Hua Teng. Spectral graph partitioning works: Planar graphs and finite element meshes. In *Proc. 37th Ann. IEEE Symp. Foundations of Computer Science (FOCS '96)*, pages 96–105, 1996.

[38] Kozo Sugiyama, Shojiro Tagawa, and Mitsuhiko Toda. Methods for visual understanding of hierarchical system structures. *IEEE Trans. Systems, Man and Cybernetics*, 11(2):109–125, 1981.

[39] William T. Tutte. How to draw a graph. *Proceedings of the London Mathematical Society, Third Series*, 13:743–768, 1963.

[40] Christopher Walshaw. A multilevel algorithm for force-directed graph drawing. In *Proc. 8th Intl. Symp. Graph Drawing (GD 2000)*, Springer LNCS 1984, pages 171–182, 2001.

[41] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of "small-world" networks. *Nature*, 393:440–442, 1998.

[42] Howard D. White and Katherine W. McCain. Bibliometrics. *Annual Review of Information Science and Technology*, 24:119–186, 1989.