Exploring the Computing Literature Using Temporal Graph Visualization*

C. Erten, P. J. Harding, S. G. Kobourov, K. Wampler and G. Yee
Department of Computer Science
University of Arizona

ABSTRACT

We present a system for the visualization of computing literature with an emphasis on collaboration patterns, interactions between related research specialties and the evolution of these characteristics through time. Our computing literature visualization system, has four major components: A mapping of bibliographical data to relational schema coupled with an RDBMS to store the relational data, an interactive GUI that allows queries and the dynamic construction of graphs, a temporal graph layout algorithm, and an interactive visualization tool. We use a novel technique for visualization of large graphs that evolve through time. Given a dynamic graph, the layout algorithm produces two-dimensional representations of each timeslice, while preserving the mental map of the graph from one slice to the next. A combined view, with all the timeslices can also be viewed and explored. For our analysis we use data from the Association of Computing Machinery's Digital Library of Scientific Literature which contains more than one hundred thousand research papers and authors. Our system can be found online at `http://tgrip.cs.arizona.edu`.

Keywords:

dynamic graph layout algorithms, computing literature visualization

1. INTRODUCTION

Information visualization techniques help in the discovery and understanding of the underlying interrelationships present in scientific literature. Techniques that capture the dynamic nature of the subject domain are especially important. In this paper, we present a system for the exploration of computing literature databases using a novel algorithm for visualization of large graphs that evolve through time. We also apply our system to the ACM Digital Library of Scientific Literature (*ACM Digital Library*) dataset which contains more than 100,000 authors and 100,000 papers. A local copy of the *ACM Digital Library*, current as of 2000, was kindly provided to us by the ACM.

An overview of our computing literature visualization system is shown in Fig. 1. The system has four major components:

1. A mapping of bibliographical data to relational schema coupled with an RDBMS to store the relational data.
2. An interactive GUI that allows queries and the dynamic construction of graphs.
3. A temporal graph layout algorithm.
4. An interactive visualization tool.

In the following sections we survey related work, present each of the main components of our system, examine relevant statistics collected from the ACM Digital Library, and consider future work.

*This work is supported in part by the NSF under grant ACR-0222920.

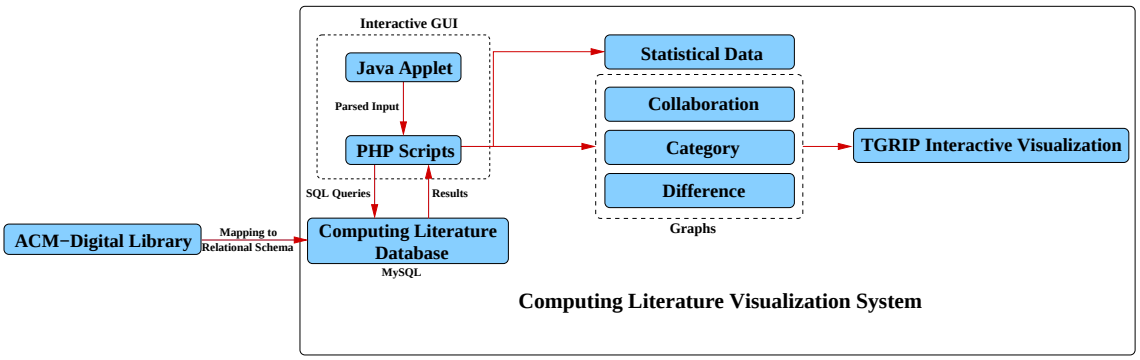


Figure 1. Overview of our system for computing literature visualization.

2. RELATED WORK

Social networks analysis, in particular scientific collaboration analysis, often relies on visualization to convey information about network structure .[2, 3, 17, 20] Author co-citation analysis has been used in infometrics and McCain[18] details the procedures required: co-citation counts are collected for pairs of authors and then stored in a co-citation matrix for further analysis. Such analysis has been applied to the library and information science domains by concentrating on the top 120 authors in the field.[25] Minimum spanning trees, based on distances between documents computed from co-citations together with multi-dimensional scaling and force-directed graph drawing methods are used to visualize parts of the information science domain.[21] Similar techniques were used to visualize the ACM Hypertext literature.[7, 8] Shneiderman *et al*[23] visualize a subset of the categorical data of the ACM Digital Library on a two axes system. GraphAEL is a system for visualization of the graph drawing literature, that allows for the exploration of co-citation and co-authorship graphs.[11]

Dynamic graph visualization is typically based on techniques for static layouts.[6, 15, 26] North[22] studies the incremental graph drawing problem in the DynaDAG system. Brandes and Wagner adapt the force-directed model to dynamic graphs using a Bayesian framework.[5] Diehl and Görg[10] consider graphs in a sequence to create smoother transitions. Most of these approaches, however, are limited to special classes of graphs and usually do not scale to large graphs. Brandes and Corman[4] present a system for visualizing network evolution in which each modification is shown in a separate layer of 3D representation with vertices common to two layers represented as columns connecting the layers. Thus, the impression that corresponding vertices appear in similar locations from layer to layer (i.e. the mental map) is preserved by precomputing locations for vertices and fixing their position throughout the layers. Collberg *et al*[9] describe a graph-based system for visualization of software evolution, which uses a modification of the GRIP algorithm for visualization of large graphs,[13] while preserving the mental map by fixing the locations of all common vertices in the evolving graph.

3. FROM DIGITAL LIBRARIES TO EXTRACTING GRAPHS

The first step in visualizing a document set is to map the dataset to a relational schema. In the case of the *ACM Digital Library* this schema (Fig. 2) contains 33 tables that represent entities and relationships, e.g., article, conference, article authored by, etc. The data is then parsed according to this schema and loaded into a MySQL database.

One of the common problems in working with a bibliographical dataset is the problem of name representation. For example, all the following are possible database entries: Edsger Wybe Dijkstra, Edsger W. Dijkstra, Edsger Dijkstra, E. W. Dijkstra, and E. Dijkstra. It is also possible that multiple distinct authors may have the same exact name. Typically these problems are addressed by choosing one way to represent the data and hoping that the resulting errors are not egregious. Excluding errors associated with distinct individuals who have the same exact name, we can establish an upper bound on the number of distinct individuals by considering every combination of first, middle and last names in the dataset as a distinct individual. In this same way we can compute a lower bound by considering every distinct combination of first initial and last name (assuming that no last names are abbreviated). In the case of the *ACM Digital Library* the upper bound is 100,988 and the lower bound is 67,189. The *ACM Digital Library* has an internal classification system for individuals where each person is assigned a distinct integer. Since it is already in place and appears to solve the problem of distinct persons with the same name, we decided to keep this internal numbering system in our database. While the data is far

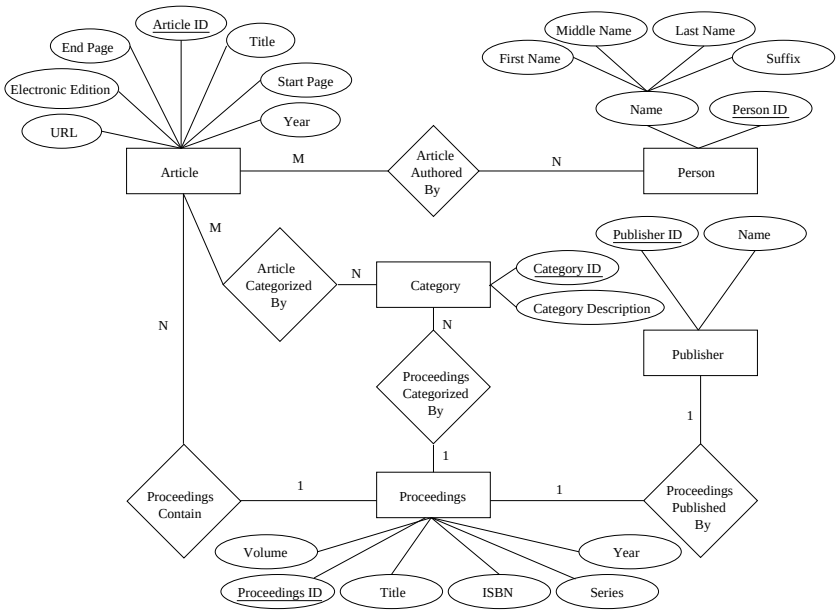


Figure 2. Abbreviated ER-Schema for the ACM Digital Library database.

from perfect, we can take comfort that errors introduced due to name representation have minor effects on overall graph statistics[20] and thus should not affect visualization.

For this particular study we use conference proceedings papers from the 20-year period from 1981 to 2000. The *ACM Digital Library* contains 51,503 conference papers and 81,279 authors in this period. Table 1 summarizes some of the important statistics gathered from the data. For years outside this range our copy of the ACM Digital Library lacks complete coverage. We decided to work with the conference data as there is better coverage and better representation of conference data in the database. We did not consider journal and conference papers together because there is non-trivial overlap of articles (journal publications that have corresponding conference versions).

From this data we generate three types of graphs. A *collaboration graph* is an undirected, vertex-weighted and edge-weighted graph representing scientific cooperation over a given period of time. Authors are represented by vertices with weight equal to that author's prominence within the document set. An edge connecting two vertices represents collaboration on a scientific paper between those two authors. The weight of the edge is equal to the frequency of collaboration. A *category graph* is an undirected vertex-weighted and edge-weighted graph representing the interconnection of subject areas based upon the built-in classification system of the *ACM Digital Library*. Vertices represent categories in the hierarchical classification system of the ACM, with weight equal to the frequency of occurrence. An edge between two vertices is drawn if a paper is classified by both categories. The weight of this edge is equal to the extent of co-occurrence in the document set. *Difference graphs* are used to visualize growth or decline between adjacent timeslices of category graphs. Vertices represent categories and have weight equal to the percentage change in occurrence from one timeslice to the next. Undirected edges with weight equal to the change in occurrence are drawn between nodes that co-occurred in both timeslices. Edges and vertices whose growth/decline is infinite have their weight adjusted to be slightly higher than the maximum weight of all other vertices that do not have infinite weight.

4. THE INTERACTIVE GUI

The interactive GUI, shown in Fig. 3 allows for the input of parameters with which to query the database and extract graphs. Large datasets, though providing potentially more voluminous and finer grained results, have the drawback of having a lower signal to noise ratio when visualizing or mining data. For this reason, users can limit results by choosing to concentrate on a specific time period, conference or conferences, group of prominent authors, or subset of categories.

Users can enter the desired parameters in a Java Applet, which then passes the information to a set of PHP scripts, which in turn dynamically query the database. The results are returned to the user in a separate window in the form of a

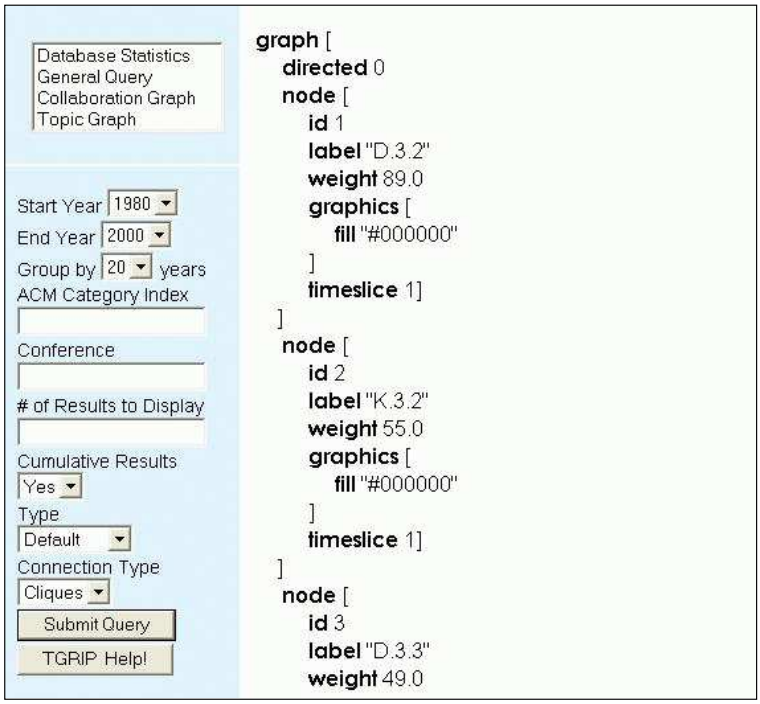


Figure 3. Snapshot of the interactive GUI for our system.

graph markup language (GML) file. Users can then invoke the layout algorithm and interactive visualization tool using the GML file.

5. THE TGRIP GRAPH LAYOUT ALGORITHM

Once graph data has been generated and returned to the user via the graphical interface, the `TGRIP` graph layout algorithm is used to calculate a suitable layout. We use a modification of `GRIP`,[13] an efficient force directed placement algorithm, to layout graphs. The essence of this algorithm is the interplay of repulsive and attractive forces. Repulsive forces exist between all vertices. Attractive forces exist wherever edges connect two vertices. The Kamada-Kawai[16] method is used to determine the layout of a sequence of filtrations, each of which is a subset of the succeeding level. Graphs in preceding filtrations are used as a guide to layout the next level. In this way large scale structures of the graph are quickly captured by initial filtration levels and small scale details are resolved in the layout of additional levels. The Fruchterman-Reingold method[12] is applied to the final filtration to achieve an even more refined layout.

Our modifications reflect the requirements of the visualization model and allow information to be represented in various ways. Vertices and edges have attributes such as weight that affect layout. For instance, in visualizing a category graph, it is useful to have a notion of both the weight of a vertex, representing overall concentration of work in that category, as well as the weight of an edge modeling the amount of interaction between categories. In addition, visualizing dynamic graphs necessitates a temporal component.

5.1. Weighted Evolving Graphs

Weights are taken into account as follows:

1. Two nodes connected by an edge of weight 0 should behave as if not connected by an edge at all.
2. An edge connecting two nodes, each of weight 0, should have a natural length of zero.
3. Heavy nodes should be placed further apart.

4. Heavy edges should be shorter.

5. If an edge of weight w connects two nodes of weight w, the edge's ideal length should be the same as an edge of weight 1 connecting two nodes of weight 1, but the larger the w, the stronger the connection should be.

Given these considerations, an edge e of weight w_e connecting nodes u, v of weight w_u, w_v, respectively, is given an ideal length of $\sqrt{w_u \cdot w_v}/w_e$. This formula will lead to a division by zero if $w_e = 0$. The resulting infinite distance is indeed the correct ideal distance for the force based calculations, since two disconnected nodes have only repulsive forces between them. In practice, however, this is undesirable and thus we ensure that all edges of weight zero are removed.

To account for the layout constraints of weighted graphs, the graph distance between two nodes is replaced with the ideal distance between the nodes. Because of the computational and space requirements of calculating the effects of all paths between two nodes, or of computing the shortest weighted path between them, an approximation is used. Let $p_1, p_2, \ldots, p_n$ be the sequence of nodes in the shortest unweighted path in G connecting two nodes, u and v. Then we define:

$$optD_G(u,v) = \sum_{i=1}^{n-1} \frac{\sqrt{w_{p_i} \cdot w_{p_{i-1}}}}{w_{e_{p_i p_{i-1}}}} \tag{1}$$

In practice this approximation works both quickly and well. The final force calculation in the modified algorithm is:

$$\vec{F}(v) = \sum_{u \in N_i(v)} \left(\frac{2\|p[u] - p[v]\|^2 \cdot (p[u] - p[v])}{(edgeLen \cdot optD_G(u,v))^2 + \|p[u] - p[v]\|^2} \right) - \sum_{u \in N_i(v)} (p[u] - p[v]) \tag{2}$$

5.2. Temporal Components

To allow for the visualization of a series of graphs which represent the evolution of a set of relationships over time we associate another attribute, called a *timeslice*, with each vertex. When visualizing the *ACM Digital Library*, each timeslice corresponds to a calendar year. All papers published at conferences in the same calendar year have the same timeslice value.

When visualizing an evolving graph two constraints need to be met. Each timeslice should have a pleasing layout within itself and the layout of consecutive timeslices should be similar, that is, vertices in one timeslice should be close to the positions of associated vertices in adjacent timeslices.

The degree to which information can be accurately construed from a graph is a measure of its *readability*. A graph that is highly readable will be easy to interpret without ambiguity. On a timeslice level, this means that nodes and edges will correlate to actual relations in the data that is being represented. Unfortunately, high individual timeslice readability can create problems in temporal graph visualization. If nodes that correspond to the same data occur in different spatial regions in a sequence of timeslices, then the viewer's conception of which nodes represent the same entities from timeslice to timeslice, (the viewer's *mental map*[19]), is destroyed. Thus readability and mental map preservation often impose conflicting requirements on the layout. We make several modifications to the layout algorithm to address this problem. First, we eliminate repulsive forces between vertices in different timeslices. Next, we arrange inter-timeslice edges between vertices that correspond to the same entities in different timeslices. For instance, in a collaboration graph the vertices representing a particular author in different timeslices would be connected by inter-timeslice edges.

In the `TGRIP` algorithm we balance mental map preservation and timeslice readability by varying the strength of inter-timeslice edges. Since vertices in different timeslices have no repulsive forces between them, the inter-timeslice edges attract each vertex towards its associated vertices in adjacent timeslices. The heavier these inter-timeslice edges are relative to vertex weights the greater the preservation of the mental map and the more readability of individual timeslices suffers. Our system allows the user to control the strength of the inter-timeslice edges, thus controlling the extent of readability and mental map preservation.

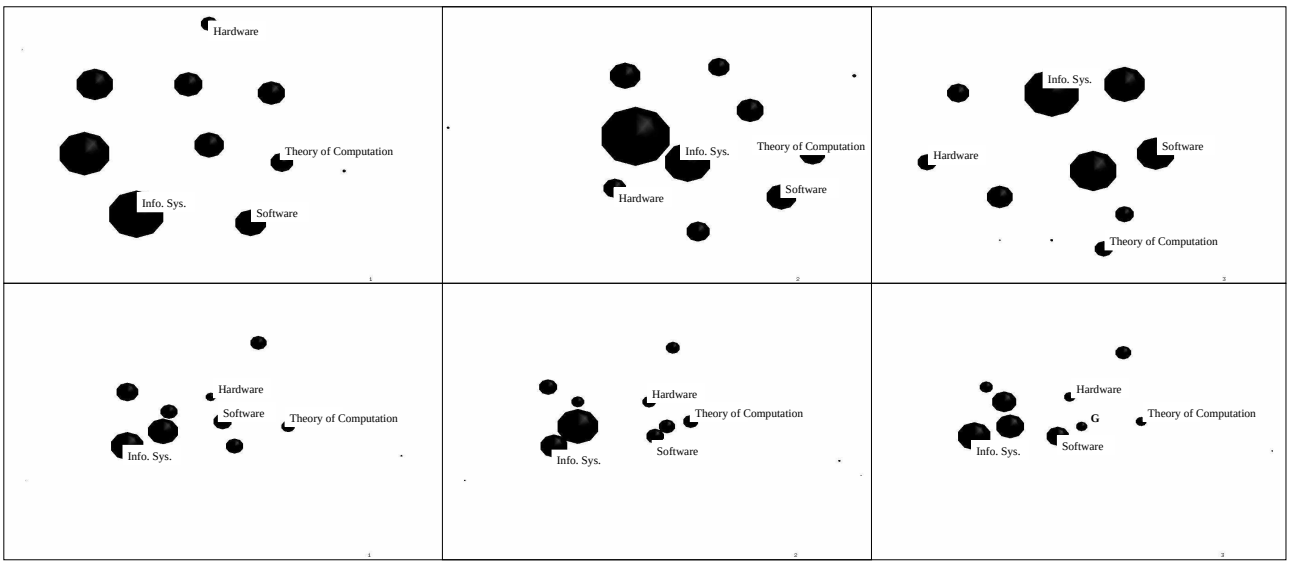


Figure 4. Top Row Level-1 category graph of three timeslices (without visible edges), from left to right, T_1 (1998), T_2 (1999), T_3 (2000) favoring readability over mental map preservation; **Bottom Row** Level-1 category graph of three timeslices (without visible edges), from left to right, T_1 (1998), T_2 (1999), T_3 (2000) favoring mental map preservation over readability;

Consider the two rows of graphs in Fig. 4. As we will examine the the placement of the vertices we have removed the edges for clarity. The layout of the individual graphs in the top row is obtained by favoring readability over mental map preservation. The layout of the individual graphs in the bottom row is obtained by favoring mental map preservation over readability. In the top row the vertices are evenly spaced and the graphs are more readable than those on the bottom. In addition, the relationships between vertices in the same timeslice are better represented. For instance, few papers are classified as being in both the Hardware and Software categories, implying that those categories are not strongly related. This can be easily verified by looking at the top row graphs which place the two vertices far apart whereas the bottom row graphs obscure this relation by placing them close to each other. On the other hand the drawings on the bottom exhibit better mental map preservation than the ones on top. In the top row the position of the vertex corresponding to the Hardware category changes drastically from one timeslice to the next, whereas the positions of the vertices are steady in the bottom row.

Mental map preservation can be accomplished by two different methods. Either all the instances of a vertex are connected via inter-timeslice edges to form a clique or each instance is only connected to the instances in its two adjacent timeslices. The first option allows global mental map preservation and is especially useful if the number of timeslices is limited. The second option is useful if the number of timeslices is large and the viewer is concerned with preserving the mental map only in a small neighborhood of timeslices.

For the combined-graph layout we constrain the drawing of time-slices to parallel planes by limiting the vertex displacement of nodes in time-slice k the plane $z = k$. We further modify the force calculations as follows: in equation (2) we re-define $optD_G(u, v)$ so that for two nodes u, v with time-slice indexes of t_u and t_v respectively:

$$optD_G(u, v) = \delta_{t_u t_v} \cdot \frac{\sqrt{w_u \cdot w_v}}{w_e}$$

where δ is the Kronecker delta ($\delta_{ij} = 1$ if $i = j$, and 0 otherwise).

6. INTERACTIVE VISUALIZATION

Once query and graph options have been input and the layout of an evolving graph has been computed, our system allows several visualization options to be employed. Edge and vertex sizes are used to represent relative weights. Timeslices

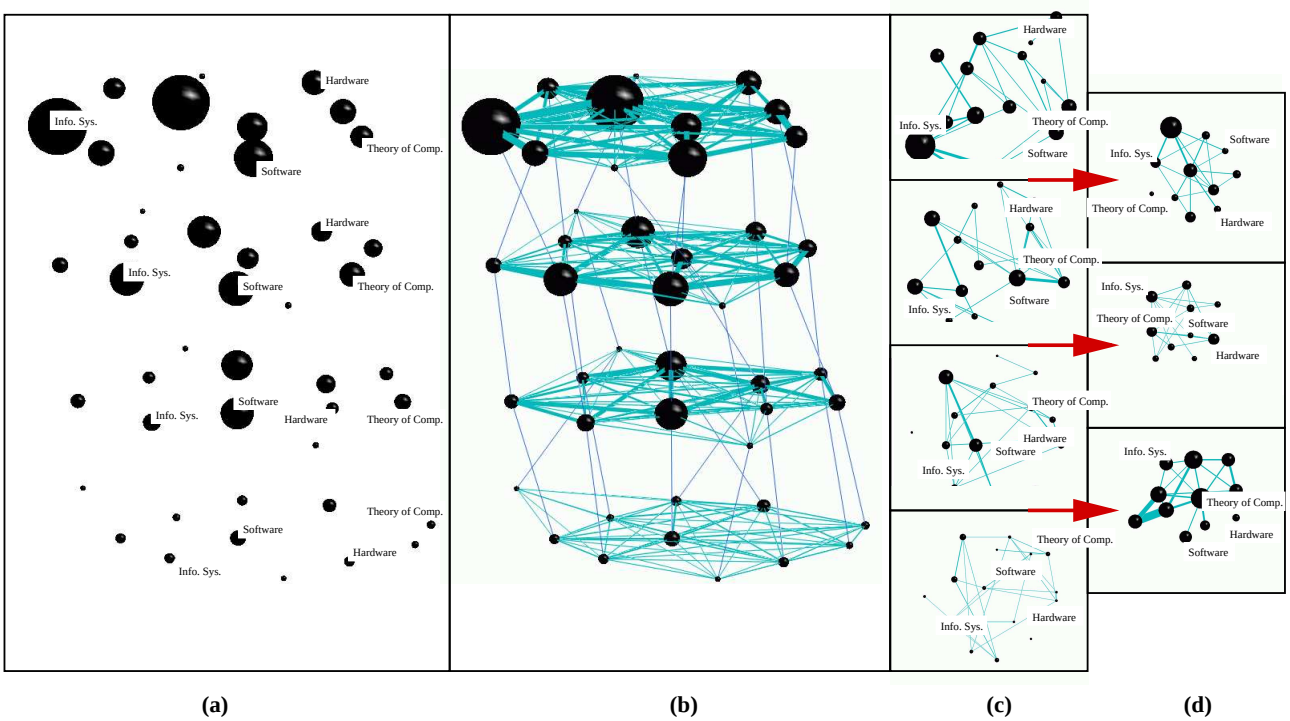


Figure 5. Level-1 category graph made of four timeslices, from bottom to top, T_1 (1981-1985), T_2 (1986-1990), T_3 (1991-1995), T_4 (1996-2000); (a) a view with invisible edges; (b) a view of the graph displaying all edges; (c) a view of the timeslices with only heavy edges; (d) a view of the difference graphs.

can be displayed in 2D or 3D, individually or together. Users have the capability to zoom in/out and to navigate in 3-dimensional space by rotating, revolving or skewing the graph. Edges can be turned invisible for better examination of vertices. In addition, evolution can be shown via an animation between timeslices in 2D or 3D. The animation uses an interpolation between timeslices and fading in/out of appearing/disappearing vertices/edges. The system, as well as static images and animation movies can be accessed at `http://tgrip.cs.arizona.edu`.

7. EXPLORING THE ACM COMPUTING LITERATURE

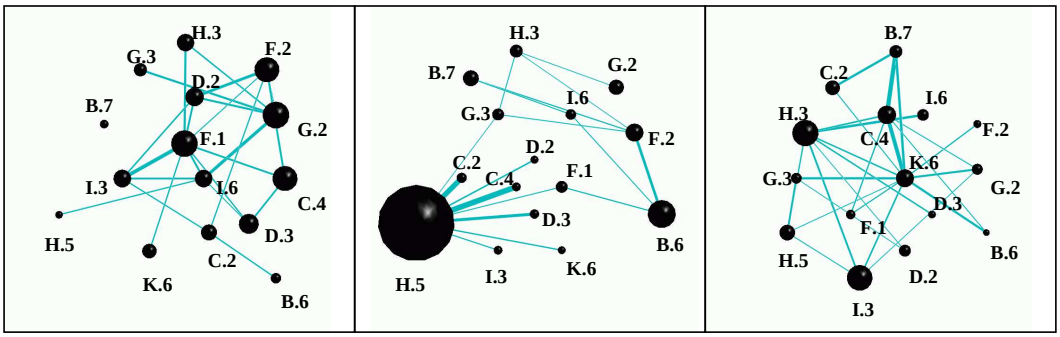
We apply our system to conference data from a copy of the ACM Digital Library, current as of 2000.

7.1. The Evolution of the Research Fields

ACM uses a hierarchical classification system to organize computing literature into 11 level-1 categories (denoted A-K) and 92 level-2 subcategories, see [24] for descriptions of these categories.

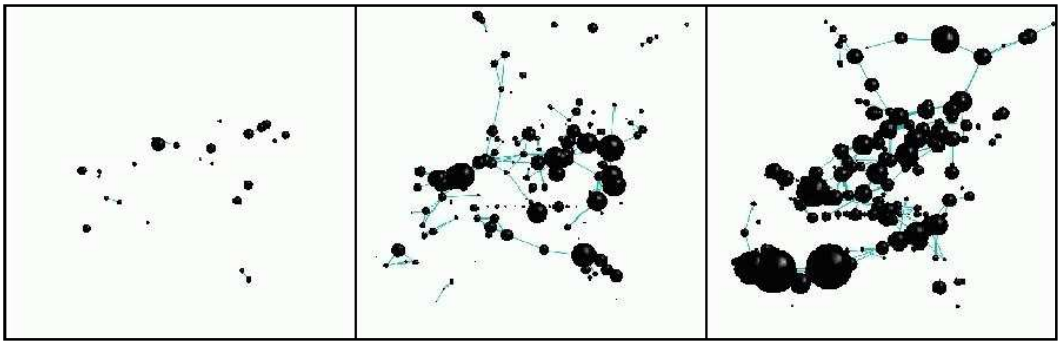
Visualizing the evolution of category graphs can reveal information about related specialties, specific concentrations of research and trends as this information evolves through time. Fig. 5 contains several visualizations of the level-1 category graph from 1981-2000 with each timeslice representing 5 years. Fig. 5(a) and Fig. 5(b) show the entire graph with invisible and with visible edges, respectively. Due to the relative weights of inter-timeslice edges, most vertices do not move a great deal between adjacent timeslices. In this example the mental map is preserved without significantly harming readability. Fig. 5(c) displays each timeslice separately in 2D. Together with category graphs, difference graphs can be effective in visualizing the evolution of the computing literature through time. Fig. 5(d) shows the difference graph corresponding to the level-1 category graph in Fig. 5(a-c). We use two different colors for vertices to distinguish between growing and declining categories. Note that although *Information Systems* is the largest category in the last time period in Fig. 5(c), its growth in the last two periods is relatively small (see Fig. 5(d)).

Once an interesting top level category is found, our system allows the user to query a subset of the lower level categories (which are more detailed and provide a better representation of research specialties) under that top level category, see Fig. 6.



(a) **(b)** **(c)**

Figure 6. Level-2 difference graphs for the following time periods; (a) Change between T_1 and T_2; (b) Change between T_2 and T_3; (c) Change between T_3 and T_4.



(a) (b) (c)

Figure 7. The H.5 (Information Interfaces and Presentation) collaboration graph: **(a)** timeslice T_1 (1986-1990); **(b)** timeslice T_2 (1991-1995); **(c)** timeslice T_3 (1996-2000)

Note the large growth of H.5 (Information Interfaces and Presentation under the upper level Information Systems category) from timeslice T_2 to T_3 in Fig. 6(b).

7.2. The Evolution of Collaborative Networks

Visualizing a collaboration graph allows a user to investigate the nature of scientific cooperation and identify interactions between research groups. As the complete collaboration graph is large and unwieldy, it is often more interesting to narrow the visualization to a certain topic.

As an example we concentrate on the level-2 category H.5 (Information Interfaces and Presentation). In order to further narrow our query we restrict the collaboration graph to the top 200 authors ranked by prominence. We define the *prominence* of an author in terms of *openness* (the number of distinct co-authors) and *productivity* (the total number of papers published). For the graph shown in Fig. 7, productivity and openness contribute equally to this ranking. This visualization can be used to investigate past and current research groups in a specific field.

In Fig. 8 we show three steps of the exploration of the graph from Fig. 7(c). While viewing this graph the user might want to focus on a particular research group. Zooming into the area highlighted by the red circle, the user can click on large vertices that seem to be central to the cluster to reveal the names of the individual authors. The clustering produced by our layout algorithm tends to group together collaborators in tight groups. For example, the cluster in Fig. 8 consists of researchers who work within H.5.2 (User Interfaces) a level-3 subcategory of H.5.

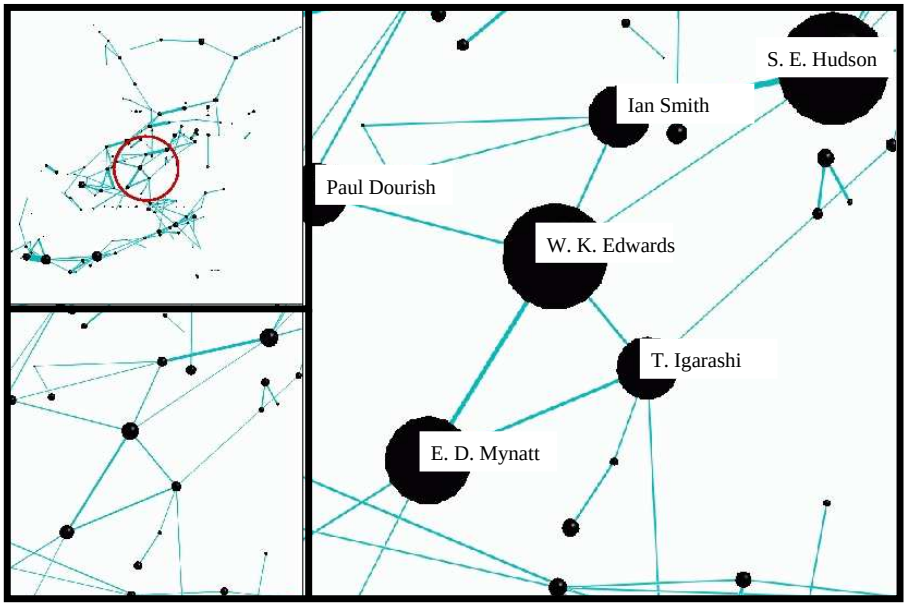


Figure 8. The H.5 (Information Interfaces and Presentation) collaboration graph: *Top left:* timeslice T_3 with smaller vertex sizes; *Bottom Left:* Zoomed into region marked with red circle; *Right:* Labeled vertices in a cluster of collaborating scientists.

8. COMPARATIVE DATA ANALYSIS AND STATISTICS

An earlier study of the computing, physics, and medical literature points to both similarities and differences between the research communities[20] in metrics such as mean number of papers and number of collaborators per author, distances between authors in the collaboration graph, and the size of the largest connected component. The data about the computing community came from NCSTRL, which contains preprints of papers submitted by participating institutions. At the time of the above study, there were slightly over than 13,000 papers and under 12,000 authors in the NCSTRL database.

We worked with the 1981-2000 conference data from the *ACM Digital Library*, which arguably presents a more complete picture of the computing literature with over 50,000 papers and over 80,000 authors. Table 1 shows a comparative summary of the overall statistics. Fig. 9 shows the cumulative number of conference papers in the period 1981-2000. The results are notable because similar data from mathematics and neuro-science[2] show linear growth while the ACM data seems to indicate super-linear growth.

General	**ACM-value**	**NCSTRL-value**
Total papers	51503	13169
Total authors	81279	11994
Authors per paper	2.32	2.22
Papers per author	1.80	2.55
Collaborators per author	3.36	3.59
Percentage of giant component	49	57.2
Percentage of 2^{nd} component	0.11	0.004
Clustering Coefficient	0.62	0.50
Average Distance	9.26	9.7
Maximum Distance	30	31

Table 1. Statistics for ACM dataset (conference papers published in 1981-2000) and NCSTRL statistics.

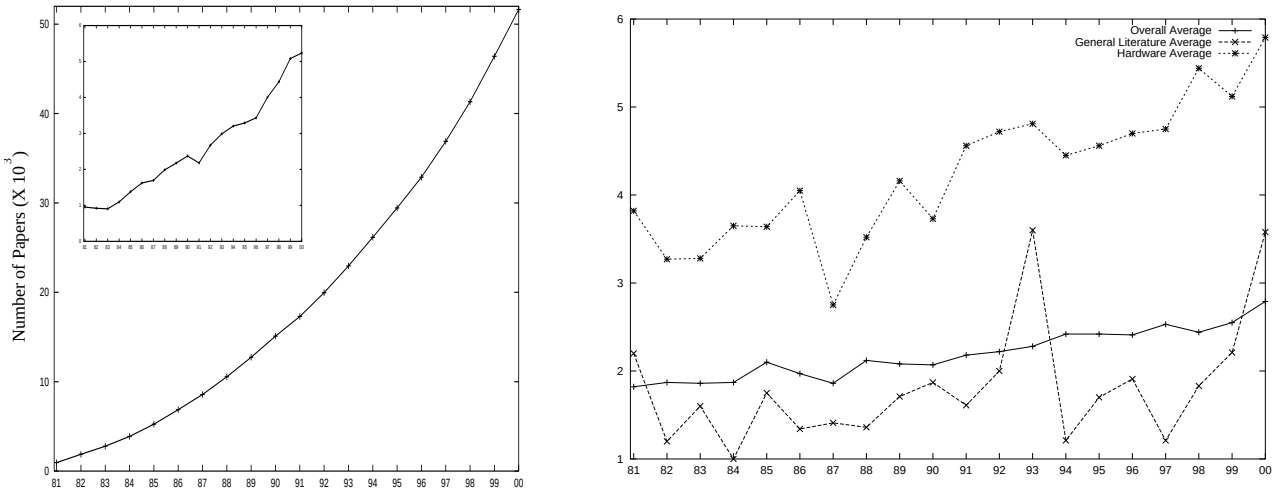


Figure 9. Left: Cumulative number of conference papers, 1981-2000 (inset shows the number of papers each year); Right: Average number of authors per paper, 1981-2000.

8.1. Authors per Paper

In mathematics, the average number of authors per paper has increased from about 1 per paper in 1935 to about 1.5 in 1995.[14] Averages in medicine and physics are often higher, about 9 collaborators per paper.[20] A steady increase in the average number of authors per paper in the computing literature (from 1.82 in 1981 to 2.79 in 2000) can be seen in Fig. 9. We include the average number of authors per paper for the overall dataset, as well as for the ACM categories with highest average, B (Hardware) and the lowest average, A (General Literature).

8.2. Size of Giant Component

In the random graph theory it is known that increasing the density of edges leads to the formation of a giant connected component. While the size of the giant component in a typical scientific collaboration graph is 80%-90% the number seems to be much smaller for the computing literature.[20] Possible reasons for the discrepancy include incomplete data and identifying one person as two or more (due to name representation). Our data indicates that the size of the giant connected component is about 49% of the overall graph. It other words, about half of the authors in the ACM dataset are connected via a path of co-authors.

8.3. Average and Maximum Distances

We can also find the shortest path from one author to another in the collaboration graph and then we can compute the average and maximum distances between pairs of vertices in the graph. The true maximum distance is infinite for two authors not in the same connected component but we perform the calculations using only the finite distances. This information is useful in the sense that it creates a chain of references of intermediate scientists through whom contact between two authors may be established.[17] The average and maximum distances in the ACM Digital Library and the NCSTRL datasets are very similar, although the former is more than three times bigger than the latter.

8.4. Clustering Coefficient

A useful measure for the strength of the ties between authors is the clustering coefficient as defined by Barabási *et al.*[2] Let N_u denote the set of neighbors of vertex u in the collaboration graph and let E_{N_u} be the set of edges e such that both of the vertices incident to e are in N_u. The clustering coefficient for u is $C_u = 2|E_{N_u}|/(|N_u| \times (|N_u| - 1))$. In other words, the clustering coefficient of u tells us how collaborative the co-authors of u are among themselves. The average clustering coefficient our dataset is 0.62, which is comparable with the clustering coefficient of the other fields such as mathematics and physics.[20]

Name	Num. of papers
Wong, D. F.	78
Cong, Jason	74
Potkonjak, Miodrag	73
Pedram, Massoud	72
Sharir, Micha	59
Shneiderman, Ben	56
Kahng, Andrew B.	56
Brayton, Robert K.	53
Sangiovanni-Vincentelli, Alberto	51
Myers, Brad A.	50

Name	Num. of co-authors
Sangiovanni-Vincentelli, Alberto	109
Shneiderman, Ben	88
Pausch, Randy	81
Fuchs, Henry	79
Soloway, Elliot	77
Kahng, Andrew B.	75
Cong, Jason	72
Druin, Allison	70
Wilson, James R.	69
Muthukrishnan, S.	69

Table 2. Authors with highest number of papers and collaborators.

8.5. Number of Papers and Collaborators

The average number of papers per author is 1.80, while the average number of collaborators is nearly double at 3.36. Table 2 shows the most productive and most collaborative authors. Changing the name representations does not affect either list significantly, with one notable exception: if all representations of Alberto Sangiovanni-Vincentelli in the ACM database are taken into account, he tops both lists. Seven of the ten researchers with highest number of papers have worked in Computer Aided Design and VLSI, two have worked in Human Computer Interaction, and one has worked in Computational Geometry.

8.6. Trends

We explored the level-2 category graphs for the 1996-2000 period. As expected, some areas (as grouped by the ACM) show decline while others seem to be growing. In particular, as shown on Fig. 10 steadily growing level-2 categories include C.5 (Computer System Implementation), E.3 (Data Encryption), H.2 (Database Management), and I.5 (Pattern Recognition). Research areas experiencing decline include E.1 (Data Structures), F.1 (Computation by Abstract Devices), and I.1 (Symbolic and Algebraic Manipulation), the last one after already experiencing a decline of 41.9% from the 1991-1995 period to the 1996-2000 period.

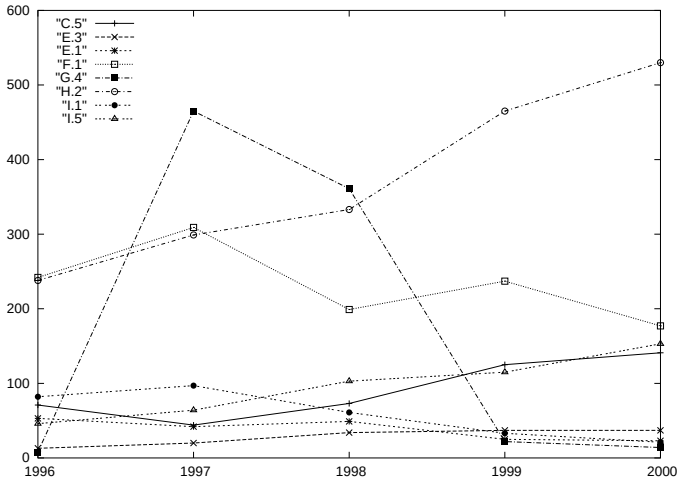


Figure 10. Growth and decline trends for C.5 (Computer System Implementation), E.1 (Data Structures), E.3 (Data Encryption), F.1 (Computation by Abstract Devices), G.4 (Mathematical Software), H.2 (Database Management), I.1 (Symbolic and Algebraic Manipulation), and I.5 (Pattern Recognition).

9. CONCLUSION AND FUTURE WORK

We have presented a system for visualization of the evolution of the computing literature using a novel graph drawing technique for visualization of large graphs with a temporal component. Category and collaboration graphs extracted from the ACM Digital Library were used to illustrate the effectiveness of the visualization model and to discover patterns and trends within the data. We provide a visual interactive tool for exploring the ACM data that we hope will be of use to the scientific community at `http://tgrip.cs.arizona.edu`.

In addition to fully integrating the current components of the system, we would like to extract citation graphs[1] and study their evolution through time. We would like to study the journal portion of the ACM database and compare and contrast it with the conference portion. We hope to be obtain a local copy of the IEEE Digital Library (for a more complete representation of the computing community) and study even larger sets using databases such as NEC's ResearchIndex.

REFERENCES

1. Y. An, J. Janssen, and E. Milios. Characterizing and mining the citation graph of the computer science literature. Technical Report CS-2001-02, Department of Computer Science, Dalhousie University, 2001.
2. A. L. Barabási, H. Jeong, Z. Néda, E. Ravasz, A. Schubert, and T. Vicsek. Evolution of the social network of scientific collaborations. *Physica*, A311:590–614, 2002.
3. V. Batagelj and A. Mrvar. Some analyses of Erdös collaboration graph. *Social Networks*, 22(2):173–186, 2000.
4. U. Brandes and S. R. Corman. Visual unrolling of network evolution and the analysis of dynamic discourse. In *IEEE Symposium on Information Visualization (INFOVIS '02)*, pages 145–151, 2002.
5. U. Brandes and D. Wagner. A Bayesian paradigm for dynamic graph layout. In *Proceedings of the 5th Symposium on Graph Drawing (GD)*, pages 236–247, 1998.
6. J. Branke. Dynamic graph drawing. In M. Kaufmann and D. Wagner, editors, *Drawing Graphs: Methods and Models*, number 2025 in LNCS, chapter 9, pages 228–246. Springer-Verlag, Berlin, Germany, 2001.
7. C. Chen and L. Carr. Trailblazing the literature of hypertext: Author co-citation analysis (1989-1998). In *Proceedings of the 10th ACM Conference on Hypertext and Hypermedia*, pages 51–60, 1999.
8. C. Chen and L. Carr. Visualizing the evolution of a subject domain: a case study. In *IEEE Symposium on Information Visualization (INFOVIS '99)*, pages 449–452, 1999.
9. C. Collberg, S. G. Kobourov, J. Nagra, J. Pitts, and K. Wampler. A system for graph-based visualization of the evolution of software. In *ACM Symposium on Software Visualization (SoftVis)*, pages 77–86, 2003.
10. S. Diehl and C. Görg. Graphs, they are changing. In *Proceedings of the 10th Symposium on Graph Drawing (GD)*, pages 23–30, 2002.
11. C. Erten, P. J. Harding, S. G. Kobourov, K. Wampler, and G. Yee. GraphAEL: Graph animations with evolving layouts. In *Proceedings of the 11th Symposium on Graph Drawing (GD)*. To appear in 2003.
12. T. Fruchterman and E. Reingold. Graph drawing by force-directed placement. *Softw. – Pract. Exp.*, 21(11):1129–1164, 1991.
13. P. Gajer, M. T. Goodrich, and S. G. Kobourov. A multi-dimensional approach to force-directed layouts. In *Proceedings of the 8th Symposium on Graph Drawing (GD)*, pages 211–221, 2000.
14. J. Grossman and P. Ion. A portion of the well-known collaboration graph. *Congressus Numerantium*, 108:129–131, 1995.
15. Herman, G. Melançon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):24–43, 2000.
16. T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Inform. Process. Lett.*, 31:7–15, 1989.
17. H. Kautz, B. Selman, and M. Shah. Referral Web: Combining social networks and collaborative filtering. *Communications of the ACM*, 40(3):63–65, 1997.
18. K. W. McCain. Mapping authors in intellectual space: A technical overview. *Journal of the American Society for Information Science*, 41:433–443, 1990.
19. K. Misue, P. Eades, W. Lai, and K. Sugiyama. Layout adjustment and the mental map. *Journal of Visual Languages and Computing*, 6:183–210, 1995.
20. M. E. J. Newman. Who is the best connected scientist? A study of scientific coauthorship networks. *Physics Review*, E64, 2001.
21. S. Noel, C. H. Chu, and V. V. Raghavan. Visualization of document co-citation counts. In *Proceedings of the Intl. Conf. on Data Mining*, pages 691–696, 2002.
22. S. C. North. Incremental layout in DynaDAG. In *Proceedings of the 4th Symposium on Graph Drawing (GD)*, pages 409–418, 1996.
23. B. Shneiderman, D. Feldman, A. Rose, and X. F. Grau. Visualizing digital library search results with categorical and hierarchical axes. In *Proceedings of the 5th ACM conference on Digital Libraries*, pages 57–66, 2000.
24. The ACM computing classification system [1998 version]. In http://www.acm.org/class/1988/.
25. H. D. White and K. W. McCain. Visualizing a discipline: an author co-citation analysis of information science, 1972-1995. *Journal of the American Society for Information Science*, 49(4):327–355, 1998.
26. K.-P. Yee, D. Fisher, R. Dhamija, and M. Hearst. Animated exploration of dynamic graphs with radial layout. In *IEEE Symposium on Information Visualization (INFOVIS '01)*, pages 43–50, 2001.