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

What are the hottest computer science research topics today? Which research areas are experiencing steady decline? How many co-authors are typical for a research paper today and 20 years ago? Who are the most prolific writers? In this paper, we attempt to address these questions as well as study collaboration patterns, research communities, interactions between related research specialties, and the evolution of these characteristics through time. For our analysis we use data from the Association of Computing Machinery's Digital Library of Scientific Literature (ACM Portal) which contains over a hundred thousand research papers and authors. 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 time-slice, while preserving the mental map of the graph from one slice to the next. A combined view, with all the time-slices can also be viewed and explored. Graphs with tens of thousands of vertices and edges, resulting from specific queries to our local copy of the ACM database, are generated and displayed in seconds. The images in this paper are produced by a graph layout tool which uses the dynamic graph layout algorithm.

Keywords:

dynamic graph visualization, visualization of literature, co-authorship analysis

1 INTRODUCTION

In this paper, we present several examples of exploration of the computing literature using a novel algorithm for visualization of large graphs that evolve through time together with more traditional gnuplot charts of the relevant statistics. The Association for Computing Machinery (ACM) has kindly provided us with a copy of the their digital library which contains over 100,000 research papers and over 100,000 unique authors. After creating our own MySQL database we wrote a program that extracts different types of graphs from the data. The graphs were then viewed with our dynamic graph visualization tool.

Typical graphs extracted from the ACM data include *category graphs* and *collaboration graphs*. The category graph allows us to view the computing literature as a large graph that evolves through time and enables us to "see" the answers to questions such as:

- What were the hottest topics in computing in the 1990's?
- Which research areas are experiencing steady decline?
- What areas have been growing rapidly in the last few years?

The collaboration graph allows us to view the computing literature as a dynamic social network and enables us to see the answers to questions such as:

- How many co-authors are typical in a research paper today?
- Which research communities are open and well-connected?
- Who is the Paul Erdös of ACM?

Several reasons influenced our decision to work with the ACM portal data: First, ACM kindly provided us with a copy of their digital library. Second, from the ACM data we were able to extract numerous graphs that evolve through time. Given access to "real-world" graphs that evolve through time we could put our dynamic graph visualization algorithm to the test. Third, we were curious to find patterns and trends within the computing community. Fourth, we were hoping to provide a visual interactive tool for exploring the ACM data that could be of use to the computing community.

2 SPECIFICS

The Data: For this study we use conference proceedings papers from the 20-year period between 1981 and 2000. The ACM Portal 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 Portal 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 a corresponding conference version). Fig. 1 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.

One of the common problems in working with a bibliographical database 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 different authors may have the same name in the database. Typically these problems are addressed by choosing one way to represent the data and hoping that the resulting errors are not that large. We did not attempt to match up different entries in the database to account for the same author appearing under more than one name. Thus, we most likely over-count the number of unique authors. Fortunately, it has been shown that errors introduced due to name representation have minor effects on the overall graph statistics [21].

Collaboration Graphs: These graphs (also known as co-authorship graphs) have been used in the past to study social networks [17] and to extract statistics about research communities [2, 21]. A well-known example of such a graph is the Erdös collaboration graph [3]. Paul Erdös was one of the most intellectually productive mathematicians in the history having authored more than 1400 papers with over 500 co-authors.

In exploring the ACM database, we would like to extract more information from a collaboration graph so that its visualization gives us a better understanding of issues such as the productivity of authors, the degree of collaboration between authors, and the evolution of collaboration patterns through time. With this in mind, our collaboration graphs are simple, undirected, node-weighted, and edge-weighted graphs corresponding to a given time period. Vertices represent unique authors and there is an edge between two vertices if the respective authors have collaborated on a research

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

General	Value
Total papers	51503
Total authors	81279
Authors per paper	2.32
Highest Category-**Hardware**	4.56
Lowest Category-**General Literature**	1.91
Papers per author	1.80
Collaborators per author	3.36
Percentage of giant component	49
Percentage of 2^{nd} component	0.11
Clustering Coefficient	0.62
Average Distance	9.26
Maximum Distance	30

Table 1: Data for conference papers published in 1981-2000.

paper. The weight of a vertex is determined by the author's collaborativeness and productivity. The weight of an edge represents the strength of the collaborative ties between two authors.

Category Graphs: We take advantage of the categorization of papers stored in the ACM database by creating what we call *category graphs*. The category graph for a given time period is a simple, undirected, node-weighted, and edge-weighted graph in which vertices correspond to categories and edges are placed between categories that co-occur in research papers. The weight of a vertex represents the concentration of research on a category and the edge weight represents the strength of the relation between two categories. There are 11 categories and 92 subcategories in the ACM classification. Related to the category graphs are the *category-change graphs* in which a vertex represents the percent change (growth/decline) of research concentration on the corresponding category. Together, the category and category-change graphs can be effectively used to visualize the trends in research specialties in computing literature through time.

Temporal Graph Visualization: Both the category and collaboration graphs contain interesting information as static representations of the computing literature given a particular year, or accumulated over a period of time. However, studying the evolution and dynamics of these graphs can reveal even more information, such as, new research trends and interesting collaboration patterns. With this in mind, we focus our attention on the models and algorithms for visualization of graphs that evolve through time (temporal graph visualization). Consider a sequence of category graphs, $G_1, G_2, \ldots, G_n$, one for each year in a given time period. To visualize the evolution of this category graph we would like to ensure that the drawing of each time-slice is *readable* and that the sequence of drawings preserves the *mental map* of the underlying structure.

Intuitively, a layout for a graph is *readable* if it shows well the relationships described by the graph structure and a user's *mental map* is preserved between two drawings in the sequence if vertices common to both graphs remain in the same positions. To address these two conflicting goals, we create a *combined-graph* $G_{1,n}$, which consist of all time-slices $G_1, G_2, \ldots, G_n$, with additional edges connecting same vertices in adjacent time-slices. We obtain a drawing for the combined-graph $G_{1,n}$ using an extensive modification of one of the algorithms for visualization of large static graphs [14]. The main features of our algorithm, `TGRIP`, include:

- graph distance based intelligent (rather than random) initial placement of vertices;
- a fast $O(n \log n + m)$ multi-dimensional scaling layout algorithm, where n and m are the number of vertices and edges in $G_{1,n}$, respectively;
- a force-directed layout for node-weighted and edge-weighted graphs, where each time-slice G_i is restricted to the plane $z = i$ and constrained by the edges connecting it to its adjacent graphs G_{i-1} and G_{i+1};

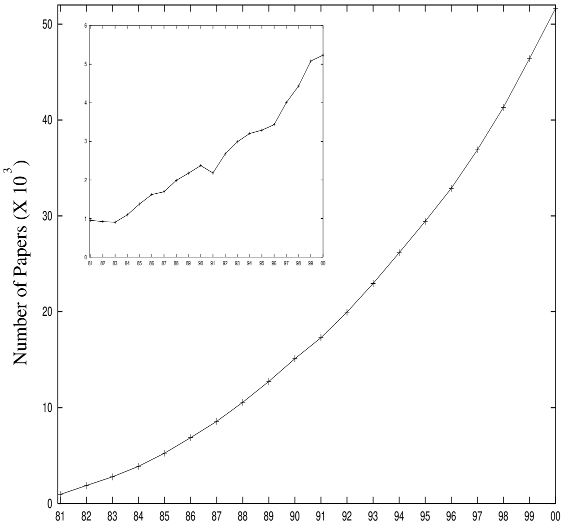

Figure 1: Cumulative number of conference papers from 1981 to 2000. The inset shows the number of papers each year.

- layout and visualization of the combined-graph, together with time-slice visualization and animation through time.
- a control mechanism for balancing readability and mental map preservation, the two main characteristics of a "good" drawing.

`TGRIP` produces a 3D drawing of the combined-graph as well as individual 2D drawings of each time-slice, as in Fig. 2(b-c). An animation of the 2D drawings of the time-slices allows us to observe the evolution of the structure.

3 Related Work

Social network analysis often relies on visualization to convey information about the network structure [24]. Social network analysis and visualization based on scientific collaborations are addressed in [2, 3, 17, 21]. Adding a visualization level to database search engines has been tried in the past. For example, Butterfly is an Information Visualizer application for accessing three DIALOG databases [18]. The Butterfly application provides a user interface for accessing multiple repositories by embedding access activity, including search and browsing within an information visualization. There have been recent efforts on mining the citation graph of the computer science literature [1] using NEC's ResearchIndex.

Author co-citation analysis has been used in informetrics and McCain [19] details the procedures required: co-citation counts are collected for pairs of authors and then stored in a raw co-citation matrix for further analysis. For instance, "maps" of a scientific domain can be generated from this data. Such analysis has been applied to the library and information science domain by concentrating on the top 120 authors in the field [25]. In [22], minimum spanning trees, based on the 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. Similar techniques were used to visualize the ACM Hypertext literature [8, 9].

Dynamic graph visualization is typically based on techniques for static layouts [6, 16, 26]. North [23] studies the incremental graph drawing problem in the DynaDAG system. Brandes and Wagner adapt the force-directed model to dynamic graphs using a Bayesian

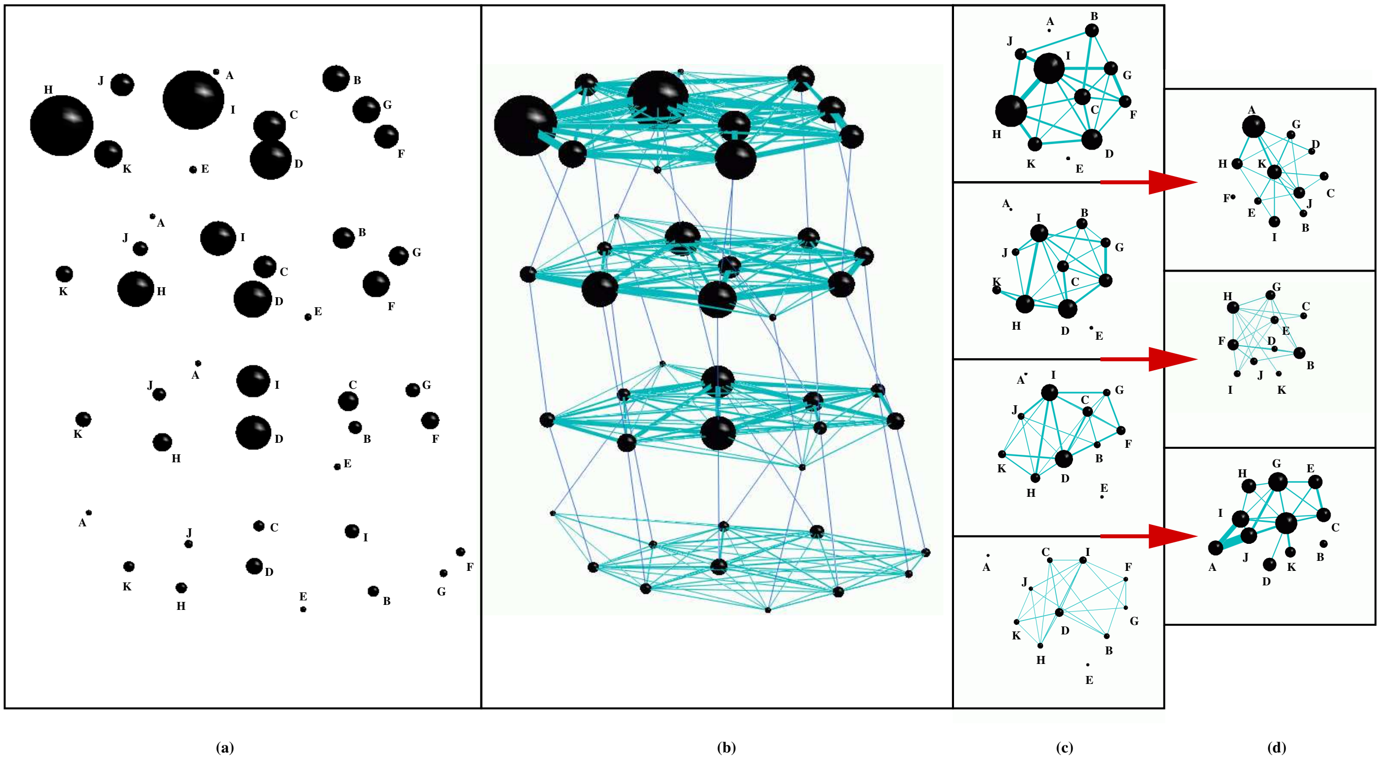

Figure 2: Level-1 category graph made of four time-slices, from bottom to top, T_1 (1981-1985), T_2 (1986-1990), T_3 (1991-1995), T_4 (1996-2000); (a) a view of the combined-graph without the edges; (b) a view of the combined-graph with all the edges; (c) a view of the time-slices with only the heavy edges; (d) a view of the category-change graphs.

framework [5]. Diehl and Görg [12] consider graphs in a sequence to create smoother transitions. Special classes of graphs such as trees, series-parallel graphs and st-graphs have also been studied in dynamic models [10, 20]. Most of these approaches, however, are limited to special classes of graphs and usually do not scale to graphs over a few hundred vertices. 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, mental map preservation is achieved by precomputing good locations for the vertices and fixing the position throughout the layers.

Simultaneous planar graph embedding is a related problem that asks whether there exist locations for the vertices of two different planar graphs such that each of the graphs can be drawn with straight lines and no crossings. Recent theoretical results [7, 13] imply that simultaneous embeddings exist only for special cases and relaxations of the problem (such as the one we address in this paper) should be considered. Along these lines, Collberg *et al* [11] describe a graph-based system for visualization of software evolution, which uses a modification of our algorithm for visualization of large graphs [14], while preserve the mental map by fixing the locations of all common vertices in the evolving graph.

4 Category Graphs

ACM classifies the computing literature in 11 level-1 categories: A (General Literature), B (Hardware), C (Computer Systems Organization), D (Software), E (Data), F (Theory of Computation), G (Mathematics of Computing), H (Information Systems), I (Computing Methodologies), J (Computer Applications), and K (Computer Milieux). Within each category there are varying numbers of subcategories, or level-2 categories, for a total of 92. The category graph for a given time period is a simple node-weighted and edge-weighted undirected graph in which vertices correspond to categories and edges are placed between categories that co-occur in research papers. The weight of a vertex in the category graph is proportional to the number of papers that list the corresponding category for the given time period. Similarly, the edge weight is proportional to the number of papers in which the two corresponding categories co-occur.

Category graphs can reveal information about related specialties, the concentration of research on a specific specialty and the trends as they evolve through time. Fig. 2 contains several visualizations of the level-1 category graph. Fig. 2(a) and Fig. 2(b) show the combined-graph without edges and with edges, respectively. The edges connecting same vertices in adjacent time-slices help with mental map preservation and are used to determine the vertex locations in the animation between time-slices. As can be seen in Fig. 2(b), most vertices do not move great deal between adjacent time-slices, thus helping preserve the mental map. Fig. 2(c) displays each time-slice separately in 2D. The evolution is animated by obtaining intermediate time-slices from any adjacent pair.

We also construct *category-change graphs* in which the weight of a vertex is the percentage of its weight change between adjacent time-slices. The edge weight is proportional to the corresponding percent-change between two time periods. Together with the category graphs, the category-change graphs can be effectively used to visualize the evolution of the computing literature through time. Fig. 2(d) shows the category-change graph corresponding to the level-1 category graph in Fig. 2(a-c). Note that although categories I (Computing Methodologies) and H (Information Systems) are the largest categories in the last time period in Fig. 2(c) the change be-

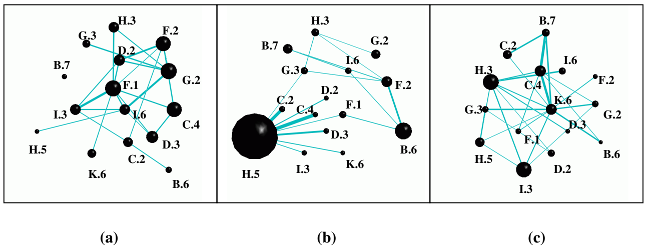

Figure 3: Level-2 category-change graphs for the following time periods; (a) Change between T_1 and T_2; (b) Change between T_2 and T_3; (c) Change between T_3 and T_4; The subcategories in the graphs are: B.6 (Logic Design), B.7 (Integrated Circuits), C.2 (Computer-Communication Networks), C.4 (Performance of Systems), D.2 (Software Engineering), D.3 (Programming Languages) F.1 (Computation by Abstract Devices), F.2 (Analysis of Algorithms and Problem Complexity, G.2 (Discrete Mathematics), G.3 (Probability and Statistics), H.3 (Information Storage and Retrieval), H.5 (Information Interfaces and Presentation), I.3 (Computer Graphics), I.6 (Simulation and Modeling), K.6 (Management of Computing and Information)

tween the last two periods is most significant in one of the smaller categories, namely A (General Literature).

We include one other example of a category-change graph for a subset of the level-2 categories, which are more detailed and provide a better representation of research specialties; see Fig. 3. Note the large growth of H.5 (Information Interfaces and Presentation) from time-slice T_2 to T_3 in Fig. 3(b).

Popular Title Words: Parsing the paper titles allows us to look for "buzzwords" in the different years. Fig. 4 shows the most popular five words (as percent of the total) used each year starting in 1981, not including words such as *for* and *the*. One noticeable trend is that compared to the early 80's there are less dominant words in the late 90's, as can be seen by the shrinking range of the plot. Some words remain quite popular throughout the 20-year period (*design, systems, simulation*) while others appear and disappear quickly (*ada, database, parallel*). The word *algorithm* makes the list in most years in the 90's and together with *model, design* and *systems* almost completes the lists for the last five years.

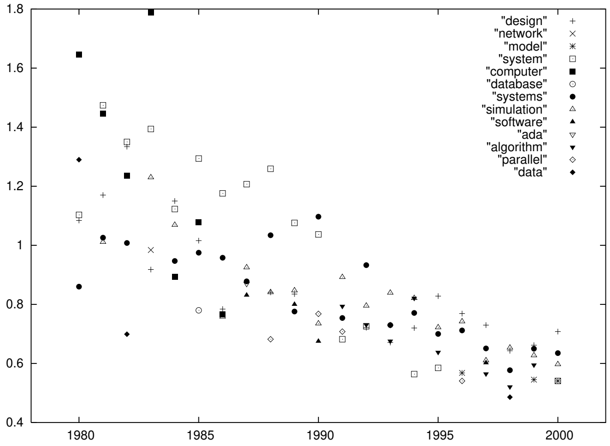

Figure 4: Title popularity

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. 5 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. Also notable is the graph for G.4 (Mathematical Software) which seems peak in 1997, followed by a sharp decline in the late 90's.

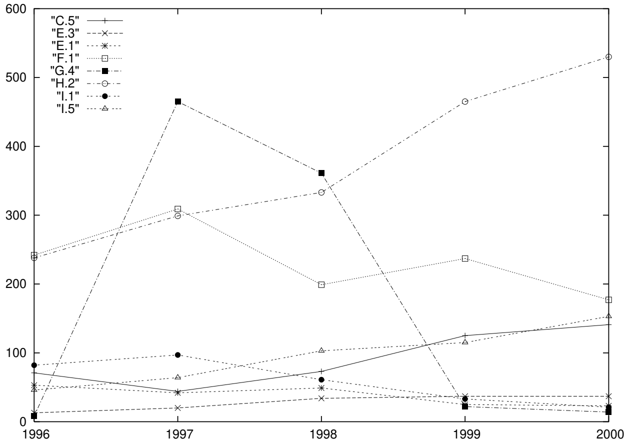

Figure 5: Growth and decline trends for C.5 (Computer System Implementation), E.1 (Data Structures), E.3 (Data Encryption), F.1 (Computation by Abstract Devices), G.4 (Mathematical Software), H.2 (Database Management), I.1 (Symbolic and Algebraic Manipulation), and I.5 (Pattern Recognition).

5 Collaboration Graphs

An earlier comparative study of the computing, physics, and medical literature points to both similarities and differences between the research communities [21] 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. However, 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 believe that our study has better coverage of the subject areas and presents a more complete picture of the computing literature. Table 1 shows a summary of the overall statistics.

Our data confirms several of the results noted in earlier papers on research collaboration as a social network [2, 21]. First, the collaboration network in computer science has the "six degrees of separation" property; that is, the average distance between two authors in the collaboration graph is a small constant. Second, as a "real-world" graph, the collaboration graph has a much higher degree of clustering than would be expected from a random graph of comparable size. Third, the power-law degree distribution in the collaboration graph places the computing literature collaboration graph in the class of scale-free networks unlike truly random Erdös-Réyni networks [24].

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 [15]. Averages in medicine and physics are often higher, with the SPIRES high-energy physics database average of about 9 collaborators per paper [21]. In computer science, theoretical papers often have less collaborators than applied papers. We note a steady increase in the average number of authors per paper

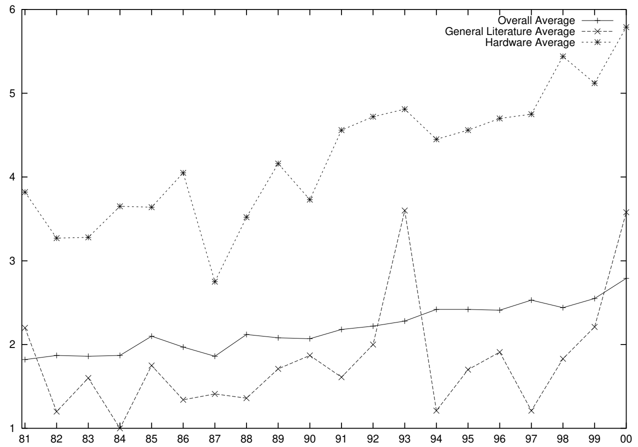

Figure 6: Average number of authors per paper.

in the computing literature from 1.82 in 1981 to 2.79 in 2000. Fig. 6 shows the average number of authors per paper in the 20-year period considered by showing the overall data, as well as data for the ACM categories with highest average, B (Hardware) and the lowest average, A (General Literature).

Size of Giant Component: In the theory of random graphs it is known that increasing the density of the 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 [21]. 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%. It other words, about half of the authors in the ACM database are connected via a path of co-authors.

Average and Maximum Distances: We can also find the shortest path from one author to another in the co-authorship graph. 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]. Our data suggests a "nine degrees of separation" property, as the average distance between two authors is slightly above 9. The maximum shortest path distance between two connected authors is 30. Note that 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.

Clustering Coefficient: A useful measure for the strength of the ties between authors is the clustering coefficient. 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 defined as:

$$C_u = 2 \times \frac{|E_{N_u}|}{|N_u| \times (|N_u| - 1)}$$

In other words, the clustering coefficient of u tells us how collaborative the co-authors of u are among themselves. We have found that the average clustering coefficient for the collaboration graph is 0.62, which is comparable with the clustering coefficient for other fields such as mathematics and physics. This indicates that the tendency to form strongly tied computing research groups is as high as in other research areas, despite the relatively small size of the giant connected component.

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 in the period 1981-2000. It is worth noting that changing the name representations did 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.

Name	Number 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	**Number 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.

Collaboration Graph Evolution: Previous work with collaboration graphs focuses on a cumulative snapshot of the graph for a given period, such as the analysis in this section. We obtained the data in Table 1 using such a snapshot of the 20-year period of interest. However, since we are interested in the dynamics of the structure, we tried to modify the traditional collaboration graph so that it captures more of the available information. For a given time period we define the collaboration graph to be a simple, undirected, node-weighted and edge-weighted graph. Vertices represent unique authors and there is an edge between two vertices if the respective authors have collaborated on a research paper. A vertex u in the collaboration graph has an *openness weight* ${w_u}^o$ equal to the number of different coauthors of the author represented by u. Similarly, u has a *productivity weight* of ${w_u}^p$ equal to the number of papers by the author represented by u. We combine these weights to form the *influential weight* ${w_u}^i = c_o \times {w_u}^o + c_p \times {w_u}^p$, where c_o, c_p are constants reflecting the importance of openness versus productivity. When visualizing the collaboration graph, the weight ${w_u}^i$ determines the size of the vertex u. The node weights play an important role in the graph layout algorithm, as node weights are used in calculating the repulsive forces between the vertices.

The weight of an edge is proportional to the number of collaborations between the respective authors in the given time period. Let k_i be the i^{th} paper of u and v together, and let n^{k_i} be the total number of authors on that paper. The weight of $e = (u, v)$ is given by $w_e = \sum_{k_i} \frac{1}{n^{k_i}-1}$. The edge weights also play an important role in the graph layout algorithm, as edge weights are used in calculating the attractive forces between the vertices.

To illustrate the algorithm and the available visualizations we look at the level-2 category H.5 (Information Interfaces and Presentation). Fig. 7 shows the top 200 (by influential weight with $c_0 = c_1$) authors in H.5 (Information Interfaces and Presentation) and their collaboration graph as it evolved through time periods T_2 (1986-1990), T_3 (1991-1995), and T_4 (1996-2000). Such visualization can be used to find out active research groups in a specific field.

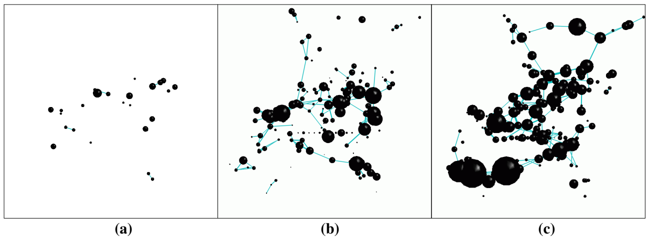

Figure 7: The H.5 (Information Interfaces and Presentation) collaboration graph: (a) time-slice T_2; (b) time-slice T_3; (c) time-slice T_4

Fig. 8 shows the collaboration graph of Fig. 7(c) in greater detail. While viewing this graph the user might want to know more about the research group clustered inside the red circle. Zooming into the area, the user can click on large vertices that seem to be central in the cluster to see the the author id's in the database. The clustering produced by the graph layout algorithm tends to group together collaborators in tight groups. For example, the cluster in Fig. 8 consist of researchers within level-3 category H.5.2 (User Interfaces) under the level-2 category H.5.

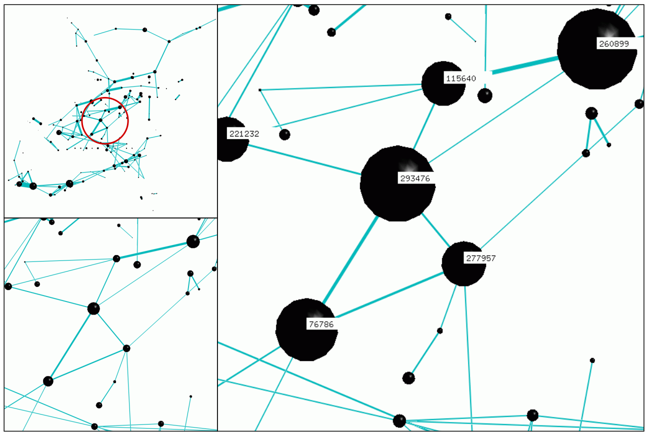

Figure 8: **Top left:** Time-slice T_4 with smaller vertex sizes; **Bottom Left:** Zoomed into region marked with red circle; **Right:** Labeled vertices in a cluster of collaborating scientists.

6 Temporal Graph Visualization

The main contribution of this paper is the algorithm for visualization of large graphs that evolve through time. We designed and implemented TGRIP, an algorithm for visualization of the combined-graph, which is similar to those in [4, 11, 14]. However, unlike earlier algorithms, TGRIP is designed for node-weighted and edge-weighted graphs and produces readable drawings while preserving the mental map between adjacent time-slices. Moreover, TGRIP provides a control mechanism for balancing readability and mental map preservation, the two main characteristics of a "good" drawing.

If we were to "optimally" draw each graph in the sequence, independent of the others, we would maximize readability at the expense of mental map preservation. On the other hand, fixing the relative positions of the vertices in each time-slice would maximize mental map preservation at the expense of readability. Instead, we create a *combined-graph* $G_{1,n}$, which consist of all time-slices $G_1, G_2, \ldots, G_n$, with additional edges connecting same vertices in adjacent time-slices.

Our algorithm is based on GRIP, an earlier force-directed algorithm for large graphs [14], and determines the placement of the vertices by repeated computation of attractive and repulsive forces. The underlying principle is that vertices repel each other, while edges prevent adjacent vertices from getting too far from each other. Thus, for a given node v, the displacement is calculated by:

$$\vec{F}(v) = \sum_{u \in N_i(v)} \left(\frac{\|p[u]-p[v]\|^2}{d_G(u,v)^2 \cdot edgeLen^2} - 1 \right) (p[u]-p[v])$$

where $p[u]$ is the position of node u, $N_i(v)$ is the the neighborhood of node v, $d_G(u,v)$ is the graph distance between nodes u and v, and $edgeLen$ is the predefined optimal edge length.

The GRIP algorithm [14] is designed to quickly compute layouts for simple, unweighted graphs with tens of thousands of vertices, without assuming any information about the underlying graphs. This makes it a good base for the visualization of graphs that evolve through time. However, before we can employ the combined-graph approach, we need to modify it so that attributes such as weights on the nodes or edges of a graph are taken into account. The meaning of the node-weight varies in the types of graphs we consider: it could be be the number of papers in the given category for category graphs, or a combination of productivity and collaborativeness weight for collaboration graphs. Higher edge-weight is typically associated with stronger connection between two nodes: number of papers that list two categories together in the category graph or number of papers co-authored by a pair of authors in the collaboration graph. The weight of the edges connecting adjacent time-slices can be varied to control the degree of mental map preservation required. For example, if the adjacent graphs are very similar then light edges suffice to keep same vertices in the same position in two adjacent time-slices.

Modification to the forces that act on the nodes are made to accommodate weights and to allow for control over the balance between the readability of each time slice and overall mental map preservation. With this in mind, 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:

$$\frac{\sqrt{w_u \cdot w_v}}{w_e} \tag{1}$$

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:

$$optD_G(u,v) = \sum_{i=1}^{n-1} \frac{\sqrt{w_{p_i} \cdot w_{p_{i-1}}}}{w_{e_{p_i p_{i-1}}}} \tag{2}$$

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

$$\begin{aligned} &\vec{F}(v) = \\ &\sum_{u \in N_i(v)} \left(\frac{2\|p[u]-p[v]\|^2 \cdot (p[u]-p[v])}{(edgeLen \cdot optD_G(u,v))^2 + \|p[u]-p[v]\|^2} \right) - \\ &- \sum_{u \in N_i(v)} (p[u]-p[v]) \end{aligned} \tag{3}$$

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 (3) 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:

$$optD_G(u,v) = \delta_{t_u t_v} \cdot \frac{\sqrt{w_u \cdot w_v}}{w_e} \tag{4}$$

Where δ is the Kronecker delta:

$$\delta_{ij} = \left\{ \begin{array}{ll} 1, & i = j \\ 0, & i \neq j \end{array} \right.$$

More details about `TGRIP` as well as additional images and animations can be found at `http://tgrip.cs.arizona.edu`.

7 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 that evolve through time were used to illustrate the visualization model and to discover patterns and trends from the data. We were hoping to provide a visual interactive tool for exploring the ACM data that could be of use to the computing community. While all three stages of the process (data extraction, graph generation, graph visualization) are working, we do not have the fully integrated and stable system that is the ultimate goal of this project yet.

In addition to 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 look for similarities and differences 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 to 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 Networkss*, 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 G. Di Battista, editor, *Proceedings of the 5th Symposium on Graph Drawing (GD)*, volume 1353 of *LNCS*, 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] P. Brass, E. Cenek, C. A. Duncan, A. Efrat, C. Erten, D. Ismailescu, S. G. Kobourov, A. Lubiw, and J. S. B. Mitchell. On simultaneous graph embedding. In preparation.

[8] 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.

[9] 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.

[10] R. F. Cohen, G. Di Battista, R. Tamassia, and I. G. Tollis. Dynamic graph drawings: Trees, series-parallel digraphs, and planar ST-digraphs. *SIAM J. Comput.*, 24(5):970–1001, 1995.

[11] 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*. To appear in June 2003.

[12] S. Diehl and C. Görg. Graphs, they are changing. In *Proceedings of the 10th Symposium on Graph Drawing (GD)*, pages 23–30, 2002.

[13] C. Erten and S. G. Kobourov. Simultaneous embedding of a planar graph and its dual on the grid. In *13th Intl. Symp. on Algorithms and Computation (ISAAC)*, pages 575–587, 2002.

[14] 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.

[15] J. Grossman and P. Ion. A portion of the well-known collaboration graph. volume 108, pages 129–131, 1995.

[16] 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.

[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] J. D. Mackinlay, R. Rao, and S. K. Card. An organic user interface for searching citation links. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 67–73, 1995.

[19] K. W. McCain. Mapping authors in intellectual space: A technical overview. *Journal of the American Society for Information Science*, 41:433–443, 1990.

[20] S. Moen. Drawing dynamic trees. *IEEE Software*, 7(4):21–28, July 1990.

[21] M. E. J. Newman. Who is the best connected scientist? a study of scientific coauthorship networks. *Physics Review*, E64, 2001.

[22] 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.

[23] S. C. North. Incremental layout in DynaDAG. In *Proceedings of the 4th Symposium on Graph Drawing (GD)*, pages 409–418, 1996.

[24] S. Wasserman and K. Faust. *Social network analysis: methods and applications*. Cambridge University Press, 1994.

[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.