Tuple – Information Visualization Publications Browser

Alex Gukov*

University of British Columbia

ABSTRACT

As publication networks grow it becomes difficult to grasp their key influences and directions by examining individual papers. Making queries on specific topics or trying to understand how various topics interact within the research field is even more challenging.

Tuple is an exploratory visualization system which provides category-based overview of publication data. Using well established visualization conventions we provide methods for gaining insight into the most influential publications, research category contents and interactions between categories (and more!). We show that Tuple is an effective solution by providing an overview and answering advanced queries about InfoVis publication data.

KEYWORDS: graph, cluster, tool, article, citation

1 INTRODUCTION

When working with a new area of research it is often difficult to grasp its influences and primary directions by examining individual papers. A high level overview can accelerate this process significantly.

One of the goals of such visualization is to help the user easily identify key authors and papers. This would allow more focused searches as well as faster familiarization with the field. In many cases, however, we need finer granularity in our research scope than just the top level area. Rather than exploring publications within the entire information visualization field, we often need to focus on specific topics such as hierarchies, graph layouts, animation methods.

The most basic question about the internal structure of a category is what its most significant papers are. A more advanced query could ask how segmented a category is. We may be interested in whether there are a small number of established papers at the source of a category or many small independent clusters. Knowing evolution of a category over time is also critical. A question that may be asked is whether there is ongoing development within it or whether the publication rate slowed down because of lack of interest.

It is often also useful to understand how various categories interact with each other. We may simply be interested in their relative size and importance, but we may also want to find out how a specific category benefited from research in other areas.

Tuple is an exploratory visualization tool designed to answer the above questions. It is tailored to the InfoVis 2004 contest dataset, but is extensible to any publication network. We process the available article text to identify the major categories. The category information is then combined with the reference connectivity to produce a node-link graph overview. Various cues such as position, size, shape and color are used to provide insights into the posed questions. Interactive controls provide detailed information on demand.

*email: agukov@gmail.com

2 DATASET

While any publication network dataset is usable we are focusing on the data from the InfoVis 2004 contest [1]. One advantage of this choice is that the data has gone through a cleanup step by at least one group of contestants. The cleaning process included removing duplicate authors, keywords as well as tying broken references. We are using the results produced by the Indiana University group [2].

2.1 Articles

The primary source of article data for this project comes from a table of 614 InfoVis articles dated from 1974 to 2004. The metadata includes title, abstract and keywords as well as publication location and date. Only 429 papers have an abstract, 424 articles have keywords and 340 have both. We do not make use of publication location as it is beyond the scope of this project.

2.2 References

There are total of 3780 references to ACM papers. Among these 1970 are to InfoVis papers within the primary article set, while 1810 are to other non-InfoVis papers. The metadata includes title, and publication date. The remaining 4722 references are to non ACM papers. The references between articles in the primary set are used to generate graph connectivity. All references are used to calculate the importance of each article in the primary set.

2.3 Article Authors

Articles are mapped to author names through a correspondence table. The table allows identification of all authors of an article in the order they are listed on the publication.

2.4 Category Data

The identification of article categories is one of the tasks of this project. The process is performed offline and loaded into the application through two tables. The first table maps articles to their corresponding categories, while the other lists category ids and their corresponding names. Total of ten major categories were found during text processing. Please see **Section 5** for details on how categories and their names were acquired.

3 PREVIOUS WORK

Similar tasks to the ones stated in the introduction have been proposed for the InfoVis 2004 contest. The contestants were required to create a static overview of the field and show its evolution.

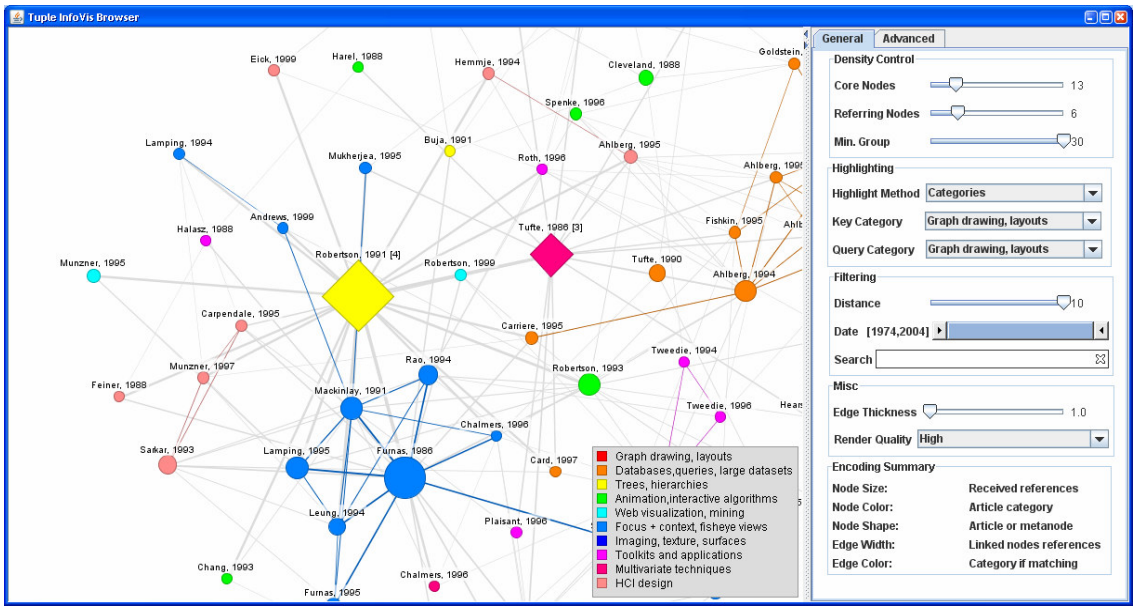


Figure 1. Tuple interface overview. The right panel displays the publication graph and the legend. The right panel contains density, highlighting and filtering controls as well as the encoding summary.

A solution from Indiana University [2] uses a node-link graph to represent a highly influential subset of the articles. The measure of importance of a particular article is the number of received references. The articles are encoded as graph nodes with their size proportional to their importance. The 'important' subset is calculated in two steps. The first step keeps only the core, the nodes with a large number of references (20 or more). The second step adds back the articles which cite the core set but have a lesser (but still significant) number of references (7 or more). This filter creates more cohesion in the resulting set, when compared to applying a direct threshold. The visualization uses citations as graph edges, allowing for easy identification of dependencies. The visualization clearly identifies the most important papers and makes it relatively easy to see the papers that cite them. However, the tight coupling of the articles in the dataset would make this type of visualization very hard to read should the number of articles increase.

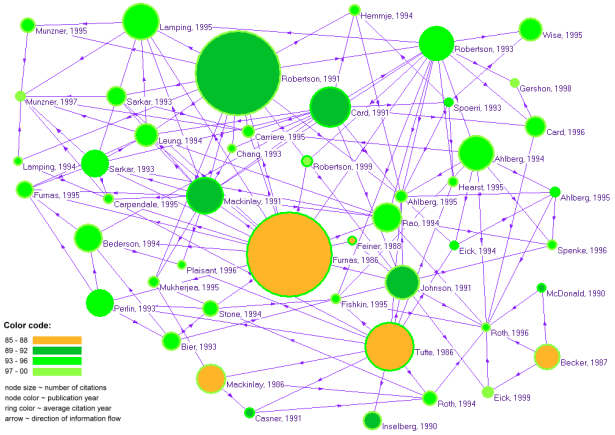


Figure 2. Indiana University InfoVis 2004 contest visualization. InfoVis paper citation network.

The authors use a different approach (burst analysis view) to visualize the evolution of the research categories within InfoVis field. Because the two views are separate, no insight can be provided into the relationship between citation network and article category groups.

A visualization from Pacific Northwest National Laboratory named IN-SPIRE [3] uses a different approach to gain insight into the data. The authors extracted key concepts from the article abstracts and projected them, along with the articles themselves onto a 2D plane. In the resulting scatter plot, the papers on a similar topic are closer to each other than papers on different topics. This is an extremely useful technique as it helps identify the relative sizes of the categories as well as the papers they include. Additionally, it provides insight into which categories are semantically close to each other. A critical advantage this visualization has over the previous example is that with more sample articles there is more insight into the data with next to no decrease in readability.

One drawback of this visualization is that it does not make use of the article citation connectivity. Because of this, there is no possibility of answering questions about the interactions of different categories. The visualization would not be helpful in finding out which categories are referenced most frequently or how the research from one category is used in another.

Most of the questions posed in the introduction require a visualization which shows citation connectivity while also providing the category information. For the reasons outlined in **Section 4.2** we chose to build our tool on the concepts developed by the Indiana University visualization.

4 SOLUTION DESCRIPTION

4.1 Overview

The solution we have implemented uses a node-link graph to provide the publication visualization. The nodes of the graph represent articles while the edges represent citations.

Only a subset of the articles is displayed. We use the same importance filtering process as the one employed by the Indiana University contest submission. As outlined in **Section 3**, the nodes are filtered by first creating the core set (articles that received a large number of citations) and the secondary set (articles with medium number of received citations which reference the core set). Because Tuple is an exploratory tool, controls have been provided to adjust graph density. The thresholds for both the core set and the secondary set can be modified resulting in an immediate graph re-layout. This is useful because large number of nodes can make a tightly interconnected graph difficult to read.

Additional insight into the data is provided by encoding information into different visualization features. Color of the nodes and edges is used to highlight categories, dates and reference direction while node size is used to indicate article importance. The specific highlighting features are controlled through the **Highlighting Mode** selection.

To aid in exploring the network we provide filtering options such as neighbour visibility, publication date range as well as text search. Unlike the density control the nodes and edges which do not fit the criteria remain visible but are faded. This aids in maintaining the mental image of the graph between searches.

Using the category and connectivity information we group similar articles into abstract meta-nodes. This feature allows for better scalability in the number of nodes and helps provide a meaningful abstract view of the research categories and their relationships.

4.2 Article citation graph

We choose a node-link graph to visualize the publications because it provides a natural way of expressing citation connectivity. We believe that it is superior to a scatter plot such as the one used in the IN-SPIRE visualization because a scatter-plot is limited to showing interactions between categories entirely through proximity. In a node-link graph we are able to use edge color and size to encode various relationships between nodes and categories. By using a specialized graph layout we are still able to leverage the position cue to emphasize node connectivity within a category.

The limited scalability of a node-link graph is addressed by providing density controls as well as grouping articles into meta-nodes.

4.2.1 Visual Encoding

Tuple's graph view uses node and edge colouring to encode features specific to each highlighting mode. However, there are a number of encoding rules which always apply.

Article nodes are represented as circles. This shape works best as it is closed, convex and does not provide any false idea of orientation [8]. Similarly to the Indiana University submission we choose node size to indicate importance. The diameter of each node is proportional to the number of received citations. Since importance is a quantitative measure, length is one of the most accurately perceived cues (second only to the position) [8]. Additionally, when the number of graph nodes is high, this technique helps direct user's attention to the more important articles first.

We use edge thickness to encode the importance of the nodes an edge is joining. This feature is useful as it highlights articles which have received a small number of references from significant articles (a different measure of importance). Additionally, it helps to identify significant article connections when a large number of overlapping edges is displayed.

4.2.2 Labeling

Labels are used to identify articles. The text of the labels is formed from the last name of the first author and the publication date (eg. Furnas, 1986). This encoding keeps the label length brief, while clearly identifying the paper.

The labels are positioned directly above article nodes. Since some of the node shapes may be small (comparable to the size of the label) positioning labels directly over the nodes, while saving space, would have significantly reduced readability. In order to always maintain text readability the labels are drawn in black over a semi-transparent white background. When the data is dense this encoding allows for the covered edges to be sufficiently visible to maintain visual continuity.

4.2.3 Basic Node Interaction

As the number of overlapping edges grows it becomes more difficult to identify article connections. To address this issue an interactive control has been added. When the user hovers the mouse over a node (article or a meta-node) the edges extending from that node are emphasized. They are drawn on top of other edges in a darker color. Additionally, the labels belonging to the neighbours of the selected node use are drawn in red.

Since the article labels provide very limited information (due to space constraints) we felt that it should be extended through the use of tool tips. This technique follows the established convention of 'details on demand' [7]. Tooltips for each article include the title, all authors and the publication date. The fields were chosen such that any ambiguity in the article identification could be resolved without consuming large amounts of screen space.

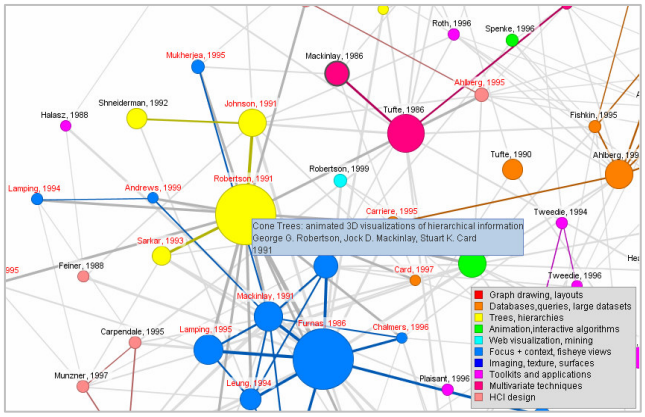


Figure 3. Node labeling and interaction. Neighbours are identified by drawing labels in red and making the edges from the selected node darker.

4.2.4 Navigation

Tuple provides standard 2D navigation options. Panning is performed by holding the left mouse button and moving the mouse in the desired direction. Zooming is performed by holding the right mouse button while moving the mouse up or down. Scrolling the mouse wheel also performs the zoom action. For convenience a fit-to-screen option is added. It can be initiated by double clicking the left mouse button. This results in a slow-in-slow-out zooming and panning animation which centers user's viewport over the graph and scales it such that all nodes are just visible.

4.3 Article Meta-nodes

When reviewing a specific category it is natural to consider clusters of connected articles as individual groups. Such a group may identify a specific idea, a collaboration circle or serve the backbone of the entire category (often the case). To clearly show interactions between these groups we represent them as single nodes. This feature also reduces the graph density, while maintaining high level information about the hidden content.

4.3.1 Construction

A meta-node is created for each cluster of two or more connected articles of the same category (Figure 5). Meta-nodes retain the category of the articles they include. The importance of a meta-node is the sum of the received reference counts of its articles. All of the search information within the grouped articles (titles and author names) is also encoded. Note that only articles which pass the density filters are involved in creation of meta-nodes (which is why we update meta nodes every time density settings change).

The label of each meta-node is constructed from the label text of the most referenced article within the group concatenated with the total number of articles (eg. Furnas 1986[22]). The tooltip text includes the title, first author and the publication date of the top five (or less) articles within the group. This reduces the need to expand/collapse the group to see its content. As some of the groups are very large (20+ articles) it is unrealistic to show them the entire content through the tooltip.

Meta-nodes inherit all connectivity of the included articles. If an article points to another article within a group an edge between the first article and the group meta-node is added. Moreover, if there is a reference between two articles in two separate groups then the two group meta-nodes are joined by an edge.

Directions carried by the grouped edges are also inherited by the edges joining meta-nodes. If articles in group A are referenced by the articles in group B then group A meta-node is referenced by the group B meta-node. Bi-directional edges are also allowed. This feature allows users to identify sources of references at the group level, which should further accelerate understanding of category interactions.

4.3.2 Encoding

In order to distinguish meta-nodes from articles we encode them using diamond shape. This is appropriate as node type is a nominal value. As in the case of article encoding, node size is proportional to the importance of the group.

4.3.3 Interaction

Interactive features of the article nodes also apply to the group meta-nodes. The user is able to highlight neighbours of a particular meta-node by hovering over it. In addition, it is also possible to expand and collapse a meta-node by pointing to it with the mouse and pressing the middle button. This allows the user to selectively reduce the amount of detail in the less interesting areas or areas which have already been examined.

4.4 Density control

Density control functions allow the user to define the importance scope of the presented articles. Three density controls are available through the options menu in addition to an interactive control.

4.4.1 Menu Controls

Modifying the reference threshold of the core set (nodes which are always included) controls the significance of the included branches. By setting the minimum core references to be a large value we force all visible articles to be derived from well established sources. On the other hand, using a smaller value we are able to explore independent groups of articles which may not have received as much exposure. Modifying the reference threshold for the secondary set (nodes which must reference the core set to be visible) we control the span of the branches formed by the core. This feature can be used to add/reduce detail of already defined groups. By default the thresholds are set to 13 for the core set and 6 for secondary set. This produces an easily readable overview of major publications. Density filtering is illustrated in Figure 4.

The minimum grouping value is the threshold which controls how large groups must be before being automatically collapsed into meta-nodes. Unlike the manual collapse/expansion this feature works on the entire graph. Note that only the nodes which remain visible after applying the reference count filters are considered for grouping. By default minimum grouping is set to 30 nodes which effectively disables it.

4.4.2 Interaction

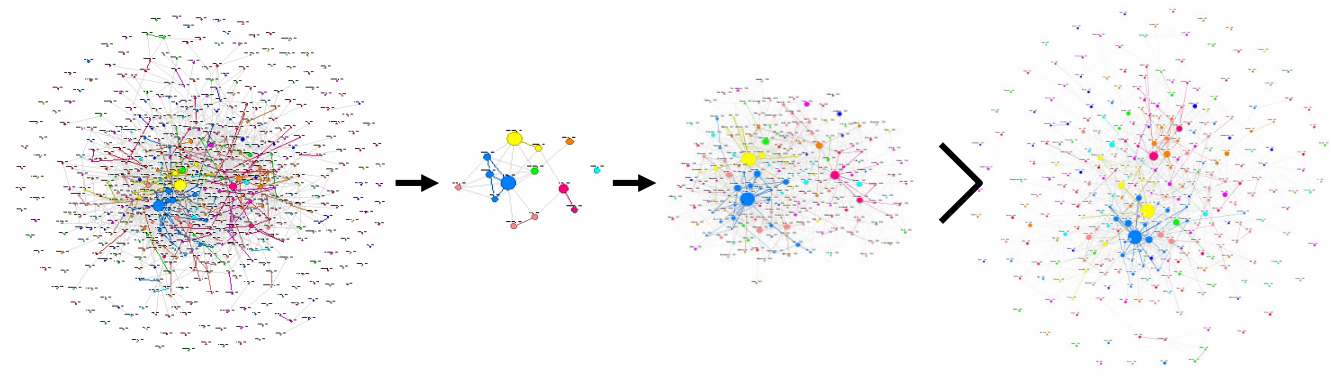
The density controls described above are effective when the user needs to change the level of detail for the entire graph. However, they do not cover the case when a specific section of the graph needs to be expanded / collapsed while keeping the remaining graph intact. This can be accomplished by directly interacting with the graph nodes. Users are able to force expansion / collapse of all neighbours of a given node by simply pointing to a node and pressing the middle mouse button in combination with CTRL.

4.4.3 Density Change Transitions

To reduce the popping effect occurring when new nodes become visible, the nodes which have previously been hidden are positioned at the mean of their visible neighbours. If fewer than three visible neighbours are available the position is unmodified (although it may be different from the last time the node was visible). If a node does not have a previous position the origin is chosen.

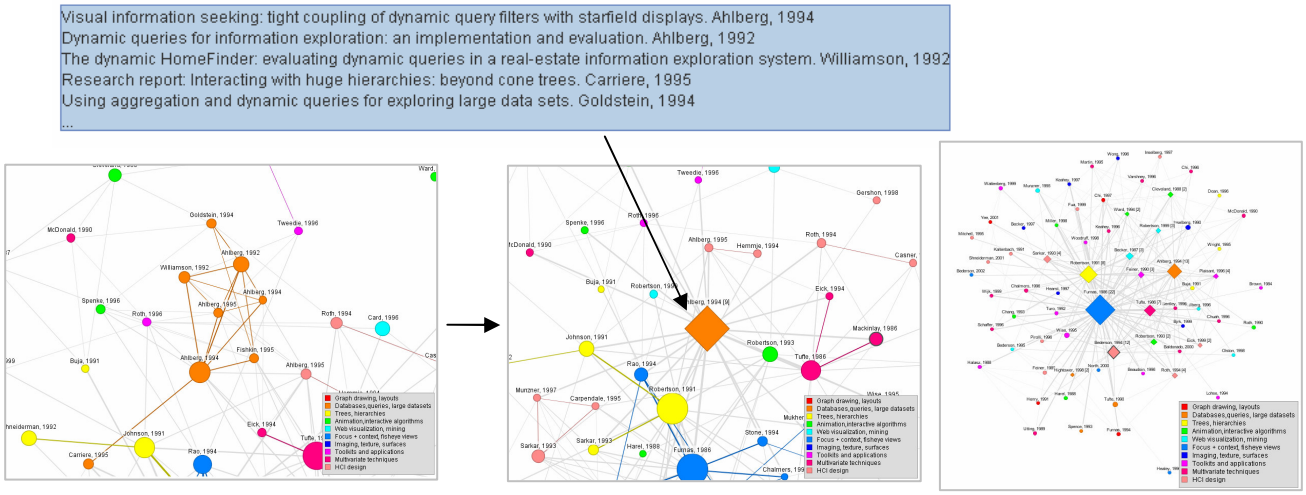
4.5 Highlighting

Using the visual encoding described so far users can easily identify the most important articles and references. Our application uses four different schemes of coloring nodes and edges to provide further insight into the data. To assist users in



(a) (b) (c) (d)

Figure 4. Density filtering process. The complete graph (a) is filtered using a high reference threshold to form the core set (b). The graph is then filtered using a medium threshold and nodes which reference the core set are included into the result (c). Using a single threshold would result in many small disconnected components (d).



(a) (b) (c)

Figure 5. Meta-node visual encoding. The meta-nodes are drawn using a diamond shape and use the most influential article as the basis for the label (b). Tooltips include descriptions of top 5 articles in the group. Meta-nodes improve scalability of the graph to more nodes. Figure 5(c) is the collapsed version of 4(c).

familiarizing themselves with the controls an encoding reference is provided in the options panel.

4.5.1 Individual Category Highlighting

Having discussed the ability to group connected articles from the same categories into meta-nodes we have not mentioned how our visual encoding helps users identify these groups. The following highlighting is designed to accomplish this task.

Initially, each category is assigned a unique color. The hue space is divided into twelve partitions and each category is assigned one of the resulting colors in sequence. If the number of categories exceeds twelve, the hues are repeated with a decreased saturation level (this does not apply to our case as we only have ten categories). Values of 1.0 (100%) are used for brightness and saturation to create maximum contrast between categories. The choice of hue to encode categories (nominal value) is consistent with established visualization conventions [8].

In order to reduce temporal mental load a legend mapping each color to its category name is shown in the bottom left corner of the screen.

Each article and group meta-node is painted using the color associated with its category. An edge is painted light gray if the nodes it is joining do not belong to the same category. In the case when the two nodes are in the same category, the edge is painted with the color associated with that category. Due to the difference in contrast between the category-colored edges and gray edges the user is able to pre-attentively identify groups of interconnected nodes.

By comparing the amount of screen space occupied by individual category colors it is trivial to discover relative sizes of the categories. By identifying the disconnected groups users are able to discover how fragmented a category is. Both of these tasks are made trivial if the groups are collapsed into meta-nodes. Examples of this highlighting mode can be found in Figures 1, 3, 4(a,b,c), 5(a,b,c).

4.5.2 Category Pair Highlighting

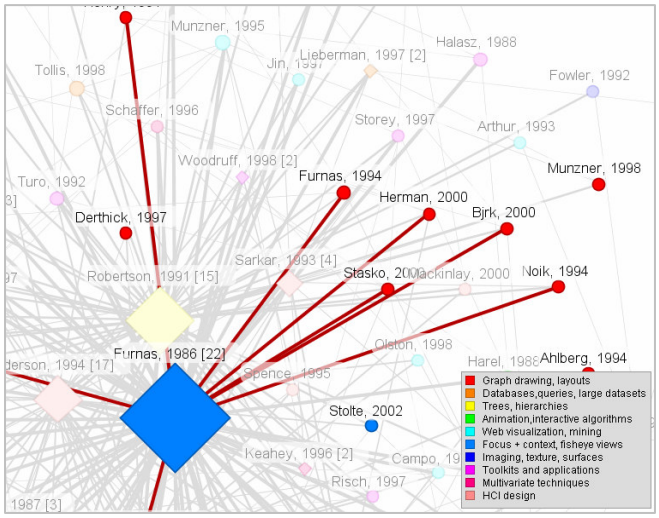


Figure 6. Category pair highlighting allows users to find relationships between categories. In this example the key category is **focus + context,** while the query category is **graph drawing, layouts**

While the previous highlighting mode allows us to explore and compare intrinsic properties of article categories, this option facilitates pair-wise category comparisons. When trying to identify results of collaboration from multiple research sub-fields a natural question is which articles in category A are related to articles in category B. A more advanced query would ask whether the coupling between categories A and B is stronger than internal coupling of A or B. Such question is useful in identifying central and peripheral categories within the research field.

For this we need the user to specify two types of categories. The first category designated as the key is probed for both internal and external connectivity. The second, query category, is compared against the key category. In Figure 6 key category is **focus+context** while the query category is **graph drawing, layouts**

Nodes are painted using their category color. Nodes which do not belong to the key or the query category are drawn faded by using transparency. The encoding is convenient, as it allows users to focus on the articles they are interested in. Edges joining nodes within the key category are painted with key category color. Edges joining nodes of the key category with the nodes of the query category are painted with the query category color. All other edges are colored in light gray.

Using the above approach we can directly compare the number of edges within the key category and the edges between key and query categories. All nodes in the query category connected to the key nodes are also easily accessible as they are (uniquely) marked by the query category edges.

4.5.3 Direction Highlighting

Previous highlighting modes do not take edge direction into account. This mode focuses on visualizing reference directions for

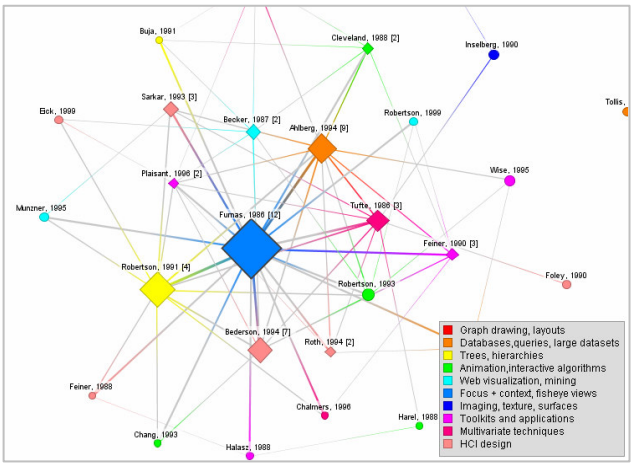


Figure 7. Reference direction highlighting allows users to identify high level relationships between categories and papers

articles and meta-nodes. By examining edges between articles users can establish which of the articles is being cited. By examining edges joining meta-nodes we can quickly make similar judgments regarding groups of articles.

As in the first two highlighting modes, nodes are painted using corresponding category colors. In order to minimize clutter we use color to represent edge direction. Using arrows would have allowed for fewer clearly visible nodes/edges.

Edges are painted using a two-color gradient. When an edge is uni-directional the gradient uses source color at the source node and grey color at the destination node. When an edge links two meta-nodes which reference each other the gradient uses two colors corresponding to the node categories.

4.5.4 Date Highlighting

The final highlighting option allows users to visually detect patterns in the article publication dates. One possible question handled by this visualization is whether there is a relationship between article publication date and the number of received references.

In order to provide this functionality we encode article publication date as node color. Because publication rate varied significantly over the 30 year span (only 15% of the articles were released in the first 16 years) we cannot use saturation or brightness to encode it. Instead we choose hue which is appropriate for high frequency data [9]. Year 1974 is mapped to hue 0.0 (red) and year 2004 is mapped to 1.0 (pink), with the remaining publication dates linearly interpolated between the two.

Meta-nodes (which do not have a date) are assigned the publication date of the first (oldest) article in the group. The value holds more information than the mean publication date, as the mean date of the highly populated groups converges to the mean of the entire data set.

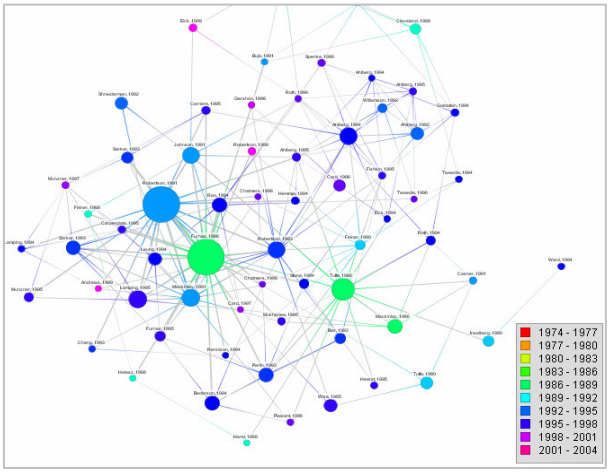


Figure 8. Visual encoding of article publication dates. The date range is mapped onto hue space. Legend available as reference.

As in the previous highlighting mode we use edge color to encode reference direction. Edges are colored with a gradient varying from the source date color at the source end to gray at the target end. If two meta-nodes reference each other the gradient transitions between two colors corresponding to their publication dates.

To reduce temporal load we display a publication date legend. Due to space limitation we do not provide the corresponding color for each individual year, but rather for each four year period.
To answer the question regarding the relationship between article importance and publication date we simply need to check if there is a relationship between node size and color (do all pink/purple colored nodes have small size ?).

4.6 Filtering

In a large graph filtering becomes critical to finding information fast. Unlike the density control, filtering does not physically remove or add nodes. This is done so that sequences of trials with wider or narrower criteria could be performed without losing a mental map of the entire graph. Instead, the nodes and edges which do not pass filtering criteria are made semi-transparent. The nodes and edges passed by the filters are colored according to the current highlighting mode.

4.6.1 Neighbour Distance Filter

This filter allows users to focus their attention on local neighbourhoods within the graph. One or more neighbourhood source must first be selected by pointing to an article or a meta-node and pressing the left mouse button. Using the distance slider it is then possible to filter out nodes that are farther than the specified number of edges from any one of the sources. While this feature is mostly used with the distance value set to one (find direct neighbours) the algorithm is generic and can handle arbitrary distances.

While there are many uses of this feature, one example is to quickly identify which articles within a meta-node are related to a given article. This is accomplished by setting the article as a neighbourhood source and reducing distance limit to one. When the meta-node is expanded all articles unrelated to the source are filtered out.

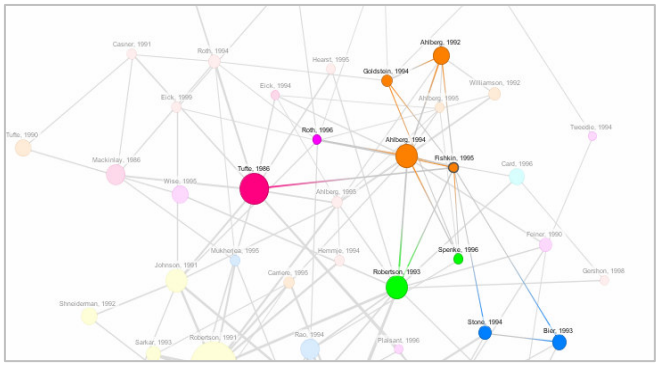


Figure 9. The distance filter can be used to limit the display to neighbours of the selected article.

4.6.2 Date filter

This filter allows users to focus on articles published within a specific date range. This is accomplished by adjusting the minimum and maximum publication year of the date slider. Once the range window has been set it is also possible to move it. This is convenient if the user is, for example, looking for a 5-year window with the most focus + context publications.

Filtering of meta-nodes is done by checking whether the minimum and maximum publication date of the grouped articles overlaps the filter window. This ensures that as long as at least one article within the group matches the criteria the meta-node is not filtered out.

4.6.3 Title and Author Search

In a dense graph it is very difficult to find a paper by a specific author. Given the fact that the title is only available as a tooltip (due to space constraints) navigating to find an article with specific title becomes impossible.

We provide text search across all article titles and author names to solve this issue. In order to find all articles by a specific author the users are able to type in the last name in the search text box and all nodes which do not match the criteria will be filtered out. The same can be performed to find articles given the title. The search is case insensitive and currently does not support advanced expressions.

5 Article Category Identification

Category identification was performed by clustering article text into semantic groups using a public domain toolkit **CLUTO** [14].

5.1 Generating Word Occurrence Matrix

For each article the source text was generated by concatenating the title, abstract and keywords. Using this vector of strings as

input we applied a Perl script **doc2mat** [15] to produce a word occurrence matrix compatible with **CLUTO**.

As a preprocessing step the script removed stop words (general use words which do not contribute to the meaning). It then applied an implementation of Porter's stemming algorithm (widely used for English language) to bring the words to common form.

The occurrence matrix was then generated by associating each unique word with a single dimension (vector index). The value at each position of an article vector was calculated to be the number of times that word occurs in the article text.

5.2 Clustering Article Text

Using the occurrence matrix created in the previous step we group the articles with **gCLUTO** application. The tool provides a number of hierarchical algorithms as well as a direct clustering algorithm k-means.

	Title	Terms
0	Graph drawing, layouts	graph, draw, layout, cluster
1	Databases,queries, large datasets	queri, databas, dynam, explor
2	Trees, hierarchies	tree, treemap, hierarchi
3	Animation, interactive algorithms	anim, algorithm, edit
4	Web visualization, mining	web, world, mine, virtual
5	Focus+ context, fisheye views	fishey, view, context, focu
6	Imaging, texture, surfaces	imag, surface, textur, field
7	Toolkits and applications	program, softwar, toolkit
8	Multivariate techniques	data, visual, sequence, analysi, multivar
9	HCI design	inform, design, interfac

Table 1. Category titles with corresponding characteristic terms

The choice of algorithm did not significantly affect the clustering results. However, our final choice was k-means as the algorithm does not impose a tree structure onto the cluster subdivision [14]. Since some of the articles are missing abstracts and/or keywords the magnitude of the vectors in the occurrence matrix varies significantly. We used cosine distance metric as it only depends on the angle between article vectors [5]. Using Euclidian distance would have produced inferior results because vector magnitude is taken into account.

After each processing run gCLUTO produces the clustering assignments as well as terms characteristic to each cluster. Additionally, we generated a data mountain visualization which uses Multi Dimensional Scaling technique to position clusters in 2D space as Gaussian peaks. Using this view we were able to judge relative distances between clusters as well as intra-cluster similarity. The mountain visualization and the characteristic terms were used to empirically determine the appropriate number of clusters. The parameters were revised until their peaks were sufficiently focused and separated and the characteristic terms for each cluster included a single high level concept. This process was completely qualitative as the division is dependent on what is considered a 'high level concept'.

After the division was performed we used the characteristic terms and information on the top articles within each category to assign meaningful category names.

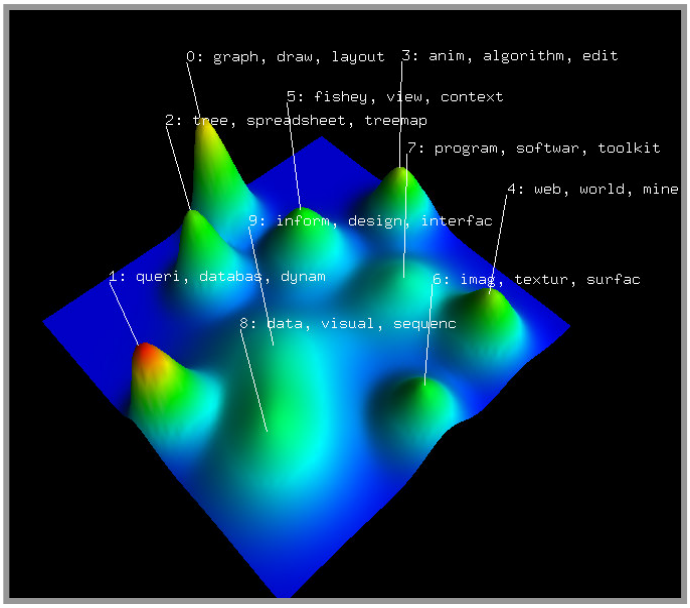


Figure 10. Data mountain visualization of category cluster distribution. Peak positions identify semantic difference. Peak height identifies internal similarity. Peak volume identifies the number of articles in the cluster.

6 Implementation

Tuple is implemented in Java [16] programming language. It is uses Prefuse [4] visualization toolkit. The category data is stored in a MySQL [17] database.

6.1 Source Data Processing

The data provided by the Indiana University InfoVis 2004 contestants was converted from MS Access to SQL format and loaded into a MySQL database. We then wrote a script which computes any additional information required for the article graph (first author, year) and imports the clustering information. The resulting data was loaded into Prefuse through MySQL-JDBC connector. The article and reference tables in the database were mapped directly to nodes and edges in the visualization graph using built in Prefuse functionality.

6.2 Visualization Application

6.2.1 Input Controls

Most settings controls such as highlighting mode, density thresholds or search criteria use standard Swing widgets. Date and text search controls were provided by Prefuse, although changes were required.

6.2.2 Filtering

Filtering is performed every time settings change. The following steps are done:

- Showing/hiding article nodes and edges given new density thresholds
- Creating meta-nodes for the current density settings (done only when density changes)
- Updating meta-node visibility depending on which groups are currently expanded
- Applying search functions to the visible nodes and marking them as pass/fail (used by visualization)

Whenever density settings change we filter the entire set of articles to produce the core and secondary set. All other articles and any edges associated with them are marked invisible.

Density change can result in completely new connected groups and we are required to recreate group meta-nodes. Category groups are generated by traversing the article graph from each node, considering only the edges joining nodes of the same category. Resulting sub-graphs with more than one node serve as bases for the meta-nodes. Any edges pointing from an article to one of the group sub-graphs is replicated to point to the corresponding meta-node. Edges between group sub-graphs result in edges between meta-nodes. Meta-node importance value, label text, date range are generated using the method described in **Section 4.3**.

Although Prefuse supports aggregate items, they do not provide sufficient functionality to implement our meta-nodes (they can't live within the graph and link to article nodes). Instead, meta-nodes are added directly to the graph and a type value is used to distinguish them from articles. Recreating meta-nodes every time density settings change works reasonably well, but an incremental update approach would have been smoother. It would have been implemented given more time.

Using the current threshold for minimum group size we adjust the visibility of the meta-nodes. If a specific meta-node becomes visible the corresponding article nodes (and their edges) are hidden.

Text search, date and neighbour distance filters are then applied to the visible nodes. The filters are simply applied in sequence marking articles and group nodes as pass or fail. The value influences how the nodes and their adjacent edges are rendered.

These functions are implemented using Prefuse GroupAction facility. All processing is custom with exception of distance and text search for which the algorithms are built in to Prefuse.

6.2.3 Layout

We make use of Prefuse force directed graph layout with customizations. The layout treats each visible graph node as a physical object which can be displaced by various forces. To add stability to the graph we set the mass of each item to be proportional to the number of received references. Thus, nodes which anchor the most neighbours are the least mobile.

Graph edges are modeled as springs between the linked nodes. By balancing nodes between their neighbours this method tries to minimize the number of edge intersections. By default all edges have equal rest length. We modify this by using a shorter rest length for the edges joining nodes within the same category. This change allows connected article groups within the same category to be positioned closer to each other.

We use anti-gravity and the drag (friction) forces without any modifications. We were only required to find the correct set of coefficients which worked with our dataset.

In order to handle disconnected components we implemented a centralization force. The force acts in the direction of the origin and is proportional to object's distance from it. Centralization force is weaker than others and has very little effect inside the core node cluster. Its benefit is that disconnected components do not float away.

6.2.4 Visualization

Node visualization is performed by a custom renderer which draws a circle or a diamond depending on the node type (article or meta-node). Color is set based on the current highlighting mode and nodes search status. Edges are drawn using a custom renderer. The color (or colors when we need a gradient) is set based on the current highlighting mode and the colors of the nodes the edge is joining.

Labels are created using Prefuse DecoratorItem facility. Each graph node is associated with its label item. The default Prefuse LabelRenderer is used. A custom layout class positions labels above their nodes. A custom ColorAction is created to specify text and background color for labels.

A custom item sorter is used to specify drawing order. Edges which are joining nodes of the same category or join key or query category nodes (in the case of **Category Pairs** highlighting) are drawn last. Since these edges are emphasized by the highlighting we need to prevent them from being covered.

6.2.5 Graph Interaction

Mouse and keyboard interaction is performed using our extension of Prefuse ControlAdapter class and the built in focus group facility. As the user clicks or hovers over items, nodes are moved in or out of designated groups. These groups include:

- Hover items. Highlight neighbour connections.
- Source nodes for neighbourhood distance filter.
- Nodes for which group expansion is forced
- Nodes for which all neighbours are forced to be shown.

The membership in each group acts as a flag for the filtering and visualization processes. We used built in Prefuse functionality to display tooltips. We were, however, required to find a custom tooltip class as the default implementation does not support multi-line text. The code for this control is borrowed from CodeGuru website [18]. We used functionality built into Prefuse for 2D navigation.

7 Scenarios of Use

This section outlines some of the practical uses of Tuple. Please refer to Appendix A for full sized screenshots of the application as it performs these tasks.

7.1 Exploring major categories

To become familiar with a research field it is useful to identify the major directions and papers. We will accomplish this by

determining top three InfoVis categories and identifying the three most influential papers in each.

We start by opening Tuple. The highlighting mode is set to **Category** and we don't need to change it. We do not need to adjust the density controls since the defaults provide a clear overview of top publications. We reduce the **Min. Group** threshold to 2. This collapses all groups into meta-nodes, thus presenting an abstract view of the data. Using our judgment of size (length cue since nodes are symmetric) we determine that the largest meta-nodes are in **focus+context**, **trees and hierarchies** and **databases, queries, large datasets**, categories. We hover over each of these meta-nodes and read the top three entries in the tooltips to determine the most influential papers.

7.2 Determining category relationships

In this scenario we want to determine which publications make practical use of hierarchical visualization techniques. There are 40 articles in the **trees and hierarchies** and 76 in the **toolkits and applications**, so brute force approach may not be very productive.

We open Tuple and select **Category Pairs** as the highlighting mode. We then choose **trees and hierarchies** as the key category and **toolkits and applications** as the query category. To see more articles we reduce the **Referring Nodes** and **Core Nodes** thresholds to zero. To make visualization clearer we collapse all groups by reducing 'Min. Group' threshold to 2.

The resulting visualization shows that the largest meta-node of the **trees and hierarchies** category is associated with 14 articles from the **toolkits and applications** category. We investigate the resulting articles by finding their abstracts on the Web (abstract display would have been a nice feature) and find four that are interesting. These include a method of visualizing stock market data using Treemaps [11], a tool for exploring complex hierarchies named Cheops [12] as well as practical evaluation studies of Treemap(rectangular) [10,13] and Sunbust (radial) [13] hierarchy visualization methods.

7.3 Publication Date vs. Importance

We are suspecting that the importance value is biased against newer publications. To confirm this we need to find out if there is a relationship between the two.

We open Tuple and select 'Dates' highlighting mode. We use the default density settings as we want to first examine the most influential papers. We can easily establish that the most influential articles have been published between 1986 and 1991. Furthermore it is clear that article date (node color) increases as the importance decreases. We setting **Date** filter window to [2000, 2004] and start reducing the minimum **Core Nodes** threshold. Using this process we find that only one paper in the last five years of the captured data has received more than two citations. This is enough to convince us that recent publications are not cited as often. This is most likely due to the fact that there has not been enough time for the references to accumulate.

8 Discussion

8.1 Strengths

The greatest strength of this visualization is that it successfully combines two information domains to provide insight unavailable when either is visualized separately. Reference connectivity is leveraged to find relationships between categories identified independently through text processing (as shown in Section 6.2).

Another key strength is the ability to provide a meaningful abstraction of groups of related nodes. Users choose the level of detail they need and are able to access further information on demand. In a tightly connected graph, such as the one we are dealing with, meta-node abstraction significantly improves readability when the level of detail is high.

We also believe that the presentation and interactivity help the application stand out. Publication details about an article and its immediate references can be easily accessed by hovering over the article node. Intuitive manipulation of graph nodes allows quick expansion of specific areas of interest. Encoding is clear and unobtrusive. As the number of graph nodes increases, category color encoding actually improves readability in what otherwise would be a soup of overlapping edges.

8.2 Weaknesses

The visualization performs well in identifying high level components (categories or groups) and their relationships. It does not, however, work well with graphs that contain many disconnected articles. As we outlined in **Section 7.3** more recent articles have received fewer citations. While this makes sense (to us), it also disconnects them from the well established networks (outgoing references do count though). The result of this is that use case in **Section 7.2** (or any case where we look for articles based their connection to a group) may be biased towards older articles. While we may consider this issue an artifact of the dataset, we can safely expect this trend to exist in any publication data.

8.3 Future Work

One of the areas we would have liked to improve is the dynamic stability of the layout. Although effort has been put in to ensure that initial positions of appearing nodes are good approximations of their final position, there are methods which can significantly improve on it. A layout algorithm described in [6] performs layout on demand (rather that continuously as we do) and uses more elaborate methods to find the initial positions of appearing nodes. Moreover, it minimizes displacement of the previously visible nodes to preserve the mental map of the graph. As this would have required significant changes to Prefuse force directed layout it was not implemented.

Another presentation component which could be further improved is the scaling of the labels. Labels are drawn in world space and at fixed scale. Because of this the labels become too small to read when the entire dataset is shown on the screen. The shortcoming is remedied by the ability to reduce data density, the tooltip mechanism as well as the ability to pan and zoom. With more time we would be able to investigate other label drawing techniques (from what is built in to Prefuse). A possible solution could have been to draw labels in screen coordinates in order of node importance, while making sure that none overlap. This way at the farthest zoom levels the labels of the most influential articles would still be clearly readable.

In addition to addressing the above (relatively minor) presentation shortcomings we would have liked to explore the relationships between various categories in the co-authorship space. As shown by the Indiana University visualization the InfoVis field started with a number of small collaboration circles,

which later merged into larger groups. We would have liked to know how these groups map to the research topics we defined, as it would provide further insight into the history of the field.

REFERENCES

[1] J. Fekete, G. Grinstein and C. Plaisant. IEEE InfoVis 2004 Contest, the history of InfoVis. www.cs.umd.edu/hcil/iv04contest (2004).

[2] K. Weimao, K. Börner and L. Viswanath. (2004). Analysis and Visualization of the IV 2004 Contest Dataset. Poster Compendium, IEEE Information Visualization Conference, pp. 49-50, 2004K.

[3] P. Wong, B. Hetzler, C. Posse, M. Whiting, S. Havre, N. Cramer, A. Shah, M. Singhal, A. Turner, J. Thomas. IN-SPIRE InfoVis 2004 Contest Entry, http://www.cs.umd.edu/hcil/InfovisRepository/contest-2004/3/unzip/PNNLstandardform2004.html

[4] J. Heer, S. Card and J. Landa. prefuse: a toolkit for interactive information visualization. In CHI Human Factors in Computing Systems, 2005.

[5] Y. Zhao and G. Karypis. Criterion functions for document clustering: Experiments and analysis. Technical Report TR #01–40, Department of Computer Science, University of Minnesota, Minneapolis, MN, 2001

[6] Y. Frishman, A. Tal: Online Dynamic Graph Drawing. EuroVis 2007: 75-82

[7] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations, In Proceedings IEEE Visual Languages , pages 336-343, Boulder, CO, Sept 1996

[8] C. Ware. Information Visualization Perception for Design. San Diego: Academic Press, 2000.

[9] B. Rogowitz and L. Treinis. How Not to Lie with Visualization, Computers In Physics 10(3) May/June 1996, pp 268-273

[10] J. Stasko, R. Catrambone, M. Guzdial, K. Mcdonald. An evaluation of space-filling information visualizations for depicting hierarchical structures, Int. J. Hum.-Comput. Stud. 53(5): 663-694 (2000)

[11] M. Wattenberg. Visualizing the stock market. In CHI '99 Extended Abstracts on Human Factors in Computing Systems (Pittsburgh, Pennsylvania, May 15 - 20, 1999). CHI '99. ACM, New York, NY, 188-189.

[12] L. Beaudoin, M. Parent, L. Vroomen. CheopsTM: A Compact Explorer For Complex Hierarchies, vis, p. 87, Seventh IEEE Visualization 1996 (VIS'96), 1996

[13] D. Turo and B. Johnson. Improving the visualization of hierarchies with treemaps: Design issues and experimentation. In Proceedings of IEEE Visualization '91, pages 124--131. Kaufman & Nielson Ed., October 1992.

[14] gCLUTO - Graphical Clustering Toolkit. http://glaros.dtc.umn.edu/gkhome/views/cluto

[15] doc2mat - Converting documents into the vector-space format used by CLUTO. http://glaros.dtc.umn.edu/gkhome/files/fs/sw/cluto/doc2mat.html

[16] Java. http://java.sun.com

[17] MySQL http://mysql.com

[18] JMultiLineToolTip. http://www.codeguru.com/java/articles.122.shtml

APPENDIX A

The following are screenshots associated with the use scenarios presented in **Section 7**.

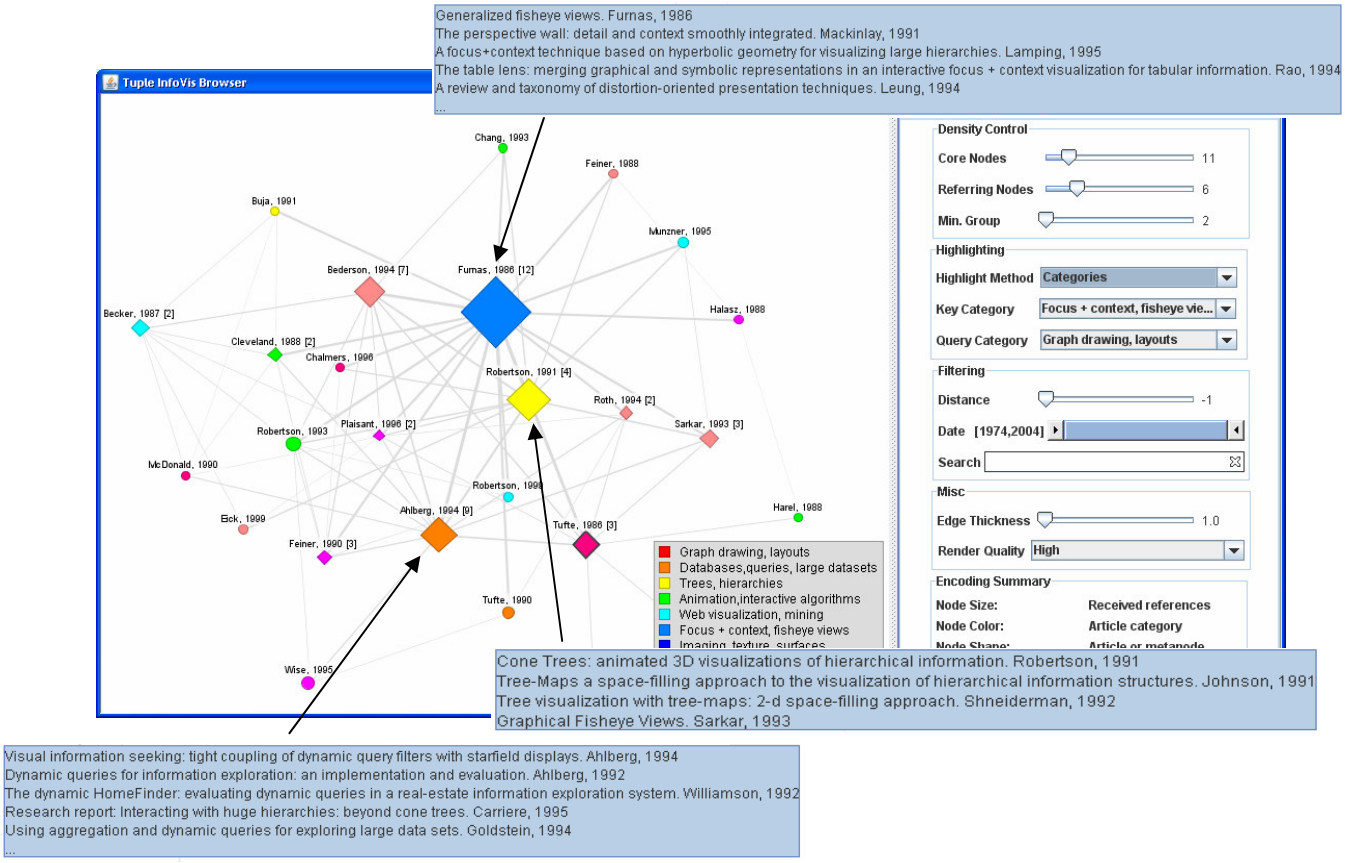


Figure 11. Identifying top categories and top publications is performed by grouping articles, identifying largest meta-nodes and reading off top entries in their tooltips (or labels if we are only interested in the top article)

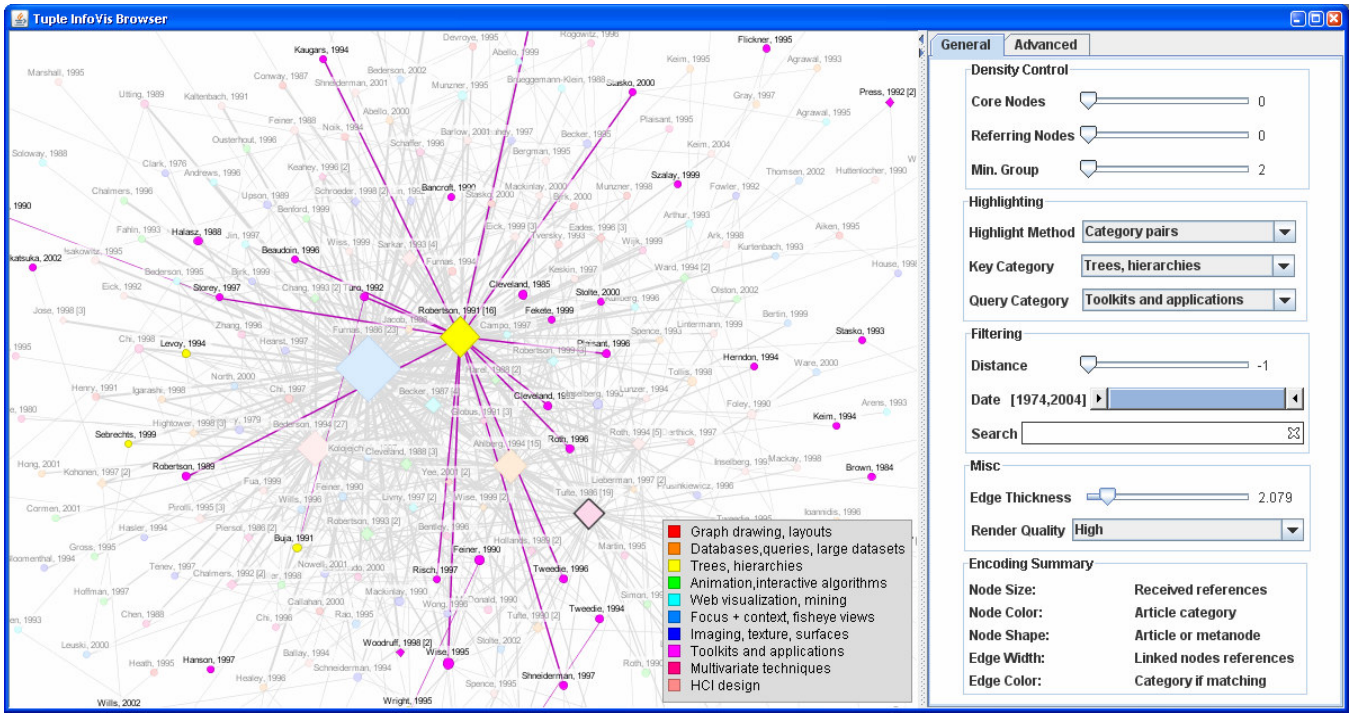


Figure 12. This view helps understand what how tree visualization techniques were used in practice. Edges are drawn from the key category(**trees,hierarchies**) nodes to the query category (**toolkits and applications**) nodes. We found 14 (out of 76 total articles) possible results (purple nodes joined to the tree category by colored edges) out of which 4 were exactly what we were looking for.

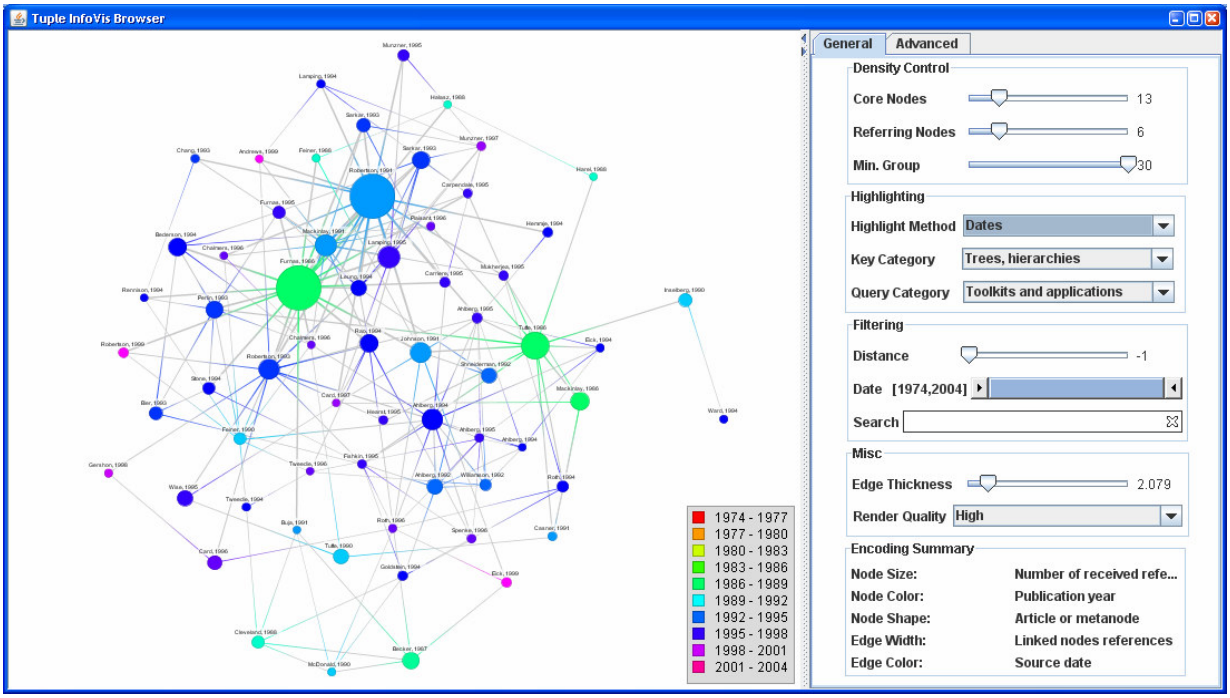


Figure 12. We use this overview to try to find a relationship between article date (color) and importance (size). After a simple glance it is clear that there are no recent nodes which have gained any significant number of references.

(a)

(b)

c)

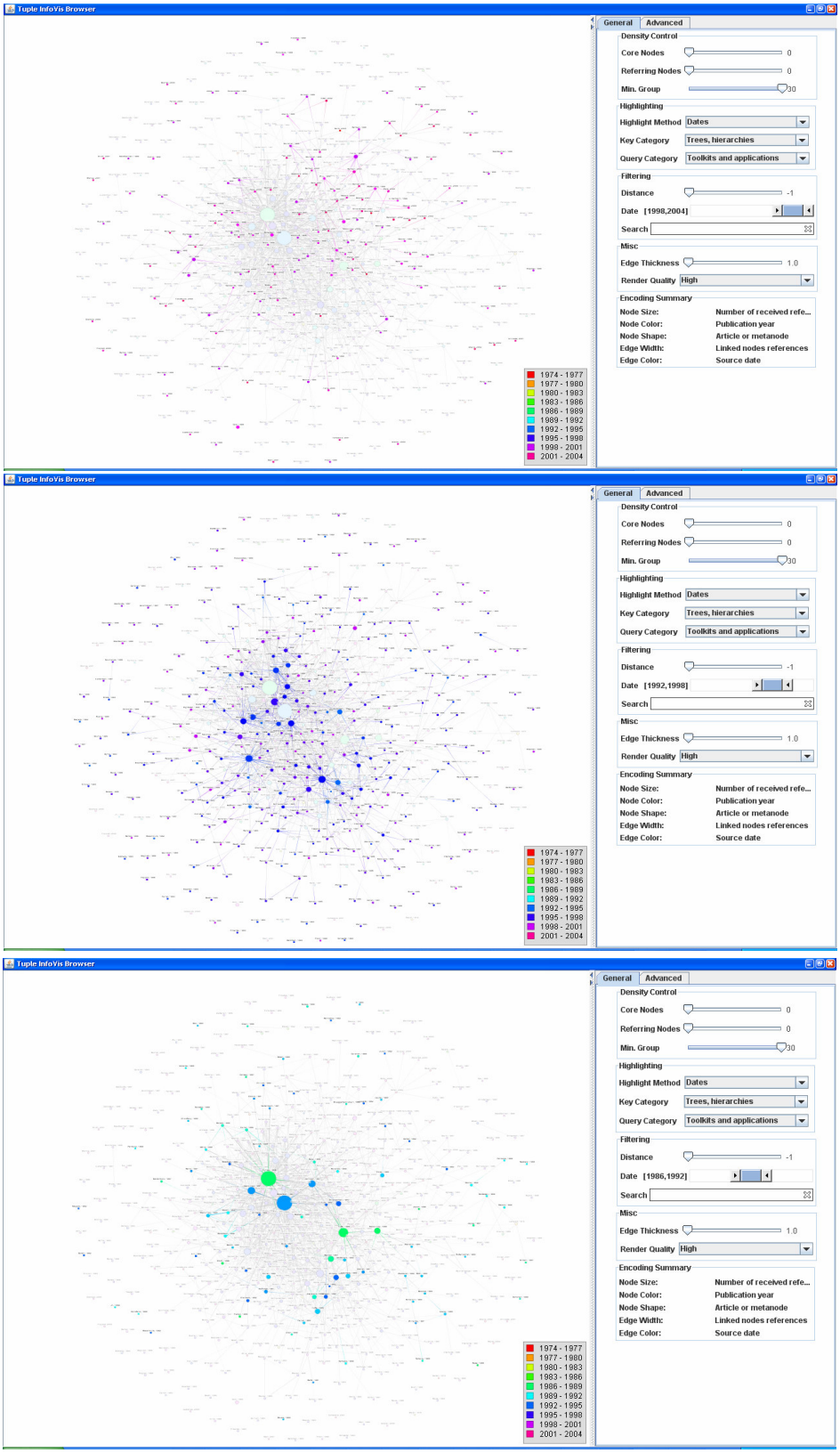


Figure 13. We use date range filter to confirm the trend between publication date and article importance. We plot filter results for three year ranges. (a) is plotted from 1998 to 2004 , (b) 1992-1998, (c) 1986-1992.