LinkScope: Interactive Graph Analysis of Unstructured Text

Yun Teng
Dept. of Computer Science,
University of California, Santa Barbara.
e-mail: yunteng.cs@cs.ucsb.edu

Tobias Höllerer
Dept. of Computer Science,
University of California, Santa Barbara.
e-mail: holl@cs.ucsb.edu

John O'Donovan
Dept. of Computer Science,
University of California, Santa Barbara.
e-mail: jod@cs.ucsb.edu

ABSTRACT

This paper presents LinkScope, a toolkit for interactive analysis of text using node link graphs, with support for dynamic addition of attributes from tabular data. The interaction technique draws on ideas from 3D modeling, mesh deformation, and static graph drawing to promote discovery of hidden information across a wide variety of graph types and analysis tasks. The key innovation of this work is the application of methods traditionally reserved for automated graph layout and clustering, to produce useful task-specific layout through dynamic interactions. Graph nodes are dynamically repositioned using an interpolated decay function over a single node movement provided by a user. We describe several variants of the interpolation method, including coupling it with a fast local-cut algorithm for cluster selection. Compared to traditional layout mechanisms the technique is particularly useful when meta-data nodes are added to a graph, increasing its connectivity. We show how the techniques can be used interactively to solve text analysis tasks including a case study on a collection of 16K awarded NSF grant proposals with meta-data and a corpus of New York Times news articles.

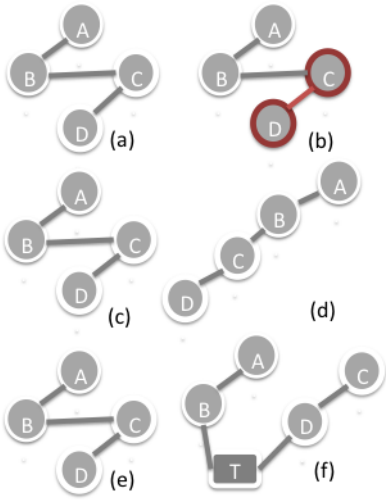


Figure 1. Example interactions in each of three classes a) to b) shows layout-preserved, c) to d) shows topology preserved, and e) to f) shows an interaction that does not preserve topology, such as the splicing of a meta-data node between two of the original nodes.

INTRODUCTION

Improvements in both hardware and software technologies are making interactive visual analytics increasingly more suitable as a solution to information overload. The key contribution of the social web –user generated content, is being produced from an abundance of sources, such as social networking applications, blogs, wikis, microblogs and digital libraries, to name a few. Many solutions to information overload have been tried and tested over the years, from the familiar index-based search engines to content-based and collaborative filters in recommender systems, collaborative search, social bookmarking services, to more structured, semantic solutions such as ontology-based search in SPARQL endpoints. All of these tools and techniques filter content in some way or other, whether passively though preference modeling, or actively through user-specified search queries. One factor that prevails is that the resulting refined information space can vary greatly in terms of content items and the interconnections that may or may not exist between them. Consider a tabular database of awarded NSF research grants for example: without advance knowledge of the database, an analyst who issues a query for "NSF program officers who awarded large grants to Californian institutions in the last year" can receive a network (of people and institutions) of arbitrary size and connectedness.

In this paper, we aim to explore this issue of uncertainty about scale and connectivity properties in filtered result sets, in particular as they apply to the process of visual analytics. To navigate a complex information space for a particular task, an analyst may construct many different types of node-link networks from an underlying data source, or set of sources, for example, by selecting attributes in a tabular structure to link entities together for a visual representation, or by adding and removing nodes and edges based on some relevance metric. This leads to an uncertainty in the complexity of the graph to be visualized for further analysis. Traditional graph layout mechanisms have inherent limitations which can render them unsuitable for some of these networks, typically as a result of scale or connectivity limitations [4].

To address this issue, we introduce a lightweight visual analytics framework which aims to support the key steps in visual data exploration: Data gathering, modeling, visualization, exploration and insight. [13]. The goal of the framework is to test the utility of two novel techniques for the interactive manipulation of node-link graphs. Both techniques are designed to work in tandem with existing graph layout algorithms, to help analysts make sense of result graphs, particularly when they become too complicated for a standalone layout algo-

rithm. Analysts can manipulate raw data in tabular form to construct graphs by selecting various column headers to link data entities derived from free text, for example, by applying topic modeling or NLP techniques. This enables the analyst to construct a broad variety of node-link graphs tailored to a given analysis task. Filters can be also applied based on attribute types in the data to further refine the graph. Once a graph has been built, automated force-based algorithms can be applied to produce graph layouts. A set of statistical analyses tools and visualizations can be used in parallel to help the analyst understand more complicated graphs, for example by examining link distributions and clusters. For the purpose of discussion and testing, we further classify this into three core areas for node-link graph manipulation: 1) Methods that preserve layout, 2) methods that preserve topology (node-link structure), and 3) methods that change the underlying node-link structure in some way.

CONTRIBUTIONS

The contributions of this paper to interactive visual analytics can be organized as follows:

- A lightweight, open-source and web-based framework for performing visual analysis of tabular, node-link and/or text-based data.
- Two novel techniques for interaction with node-link graphs. One based on interpolation of mouse gestures over the graph, and a second technique that applies EvoCut [3] conductance-based clustering and other methods prior to gesture interpolation.
- An evaluation of the interactive analysis capabilities of the framework and the interaction techniques on a set of NYT news article, a corpus of awarded NSF grants from 2008 to 2010, and on a database of terrorism-related deaths during the Northern Ireland "troubles" from 1972 to 1985.

The remainder of this paper is organized as follows. First, we will present an overview of the analysis toolkit, using an example work flow. The following section ends with a discussion of the scope of different workflows that are available. Next, we describe our two novel graph interaction algorithms in detail, including a discussion of computational complexity for each. Finally we evaluate the analysis toolkit with particular focus on the role of the interaction algorithms in the three usage scenarios.

LINKSCOPE TOOLKIT

Figure 2 shows a snapshot of the interface during an example analysis session. In this session, an NSF executive is interested in finding out the various ways the foundation is funding work related to counter-terrorism. The data shown is a set of awarded NSF grant documents from 2008 to 2010 inclusive. The main window shows the current graph view. On the left of the screen is a provenance view, where the smaller graphs depict the state of the visualization at each preceding filtering step. Each view can be reverted to at the click of a button. The fan-like panels on the right control a variety of data mining and visualization algorithms which can take selected data from the graph view as input.

The top left view shows the initial graph of 6000+ connected entities from the NSF data set. Most of these nodes represent grant documents, and they are connected through a small percentage of "topic" nodes. These nodes have been computed using Latent Dirichlet Analysis over the contents of each document, using the algorithm outlined in [6]. The layout has been produced by running a simple multidimensional scaling over the set of inter-topic similarity values output by the algorithm, thereby placing similar topic nodes in close proximity to each other. Each document/grant node has been colored based on its NSF category. For example, network science, graphics, cyber-trust, visualization etc. The clusters of color arise from the inherent topic similarity across the set of grant documents. We believe that this provides a good overview reference to begin our analysis workflow.

Next, a simple text-based query was issued over all documents for the keyword "counter-terrorism". For simplicity in this example, the search was limited to document abstracts only. The search returned a list of ten documents in a panel on the right. The analyst checks all of them and they appear on the graph as isolated nodes. At this point, a number of expand algorithms can be run to search the neighborhood of these nodes in the topic graph. In this case, the analyst is interested in a more targeted search, so, using the "tabular data" panel on the right, she opts to build edges based on the AWARDEE_STATE property. This results in the second view, which is a set of largely disconnected clusters, but with the addition of new nodes on the graph to represent the connecting attribute. The analyst can optionally apply filters on the connecting attributes at this point. For example, to view only those grants related to CA and VA states. A list of active filters is shown at the bottom of the panel. After each addition to the graph, a layout algorithm (in this example a simple Fruchterman-reingold algorithm, but optionally an FM3 [11], or binary stress layout) is automatically invoked to produce a clearer view of the new graph. The analyst repeats this process by adding edges for PROGRAM_OFFICER and COUNTRY, and the graph becomes much more connected, as shown in the remaining provenance views on the left. Edges and filters can be applied for any attribute or attribute value in the data set.

In the last provenance view, the graph has already become fairly connected, and the force directed layout is already losing clarity due to crossed edges etc. At this point, the analyst notices a seemingly prominent node (Leyland M. Jameson), and would like to investigate further. The analyst selects the node and drags it in an arbitrary direction, automatically invoking an interpolation algorithm which applies the gesture to all other nodes in the graph with a decaying weight based on hop distance from the moved node. For visual landmarking, a smooth animation is applied over the node transitions, and the graph morphs into the view shown in the main window. This tree-like structure pivots nodes around the target, and provides a clearer view of its relations. For instance, Leyland M Jameson was program officer on NSF grants related to counter-terrorism in three states: California, Texas and Michigan.

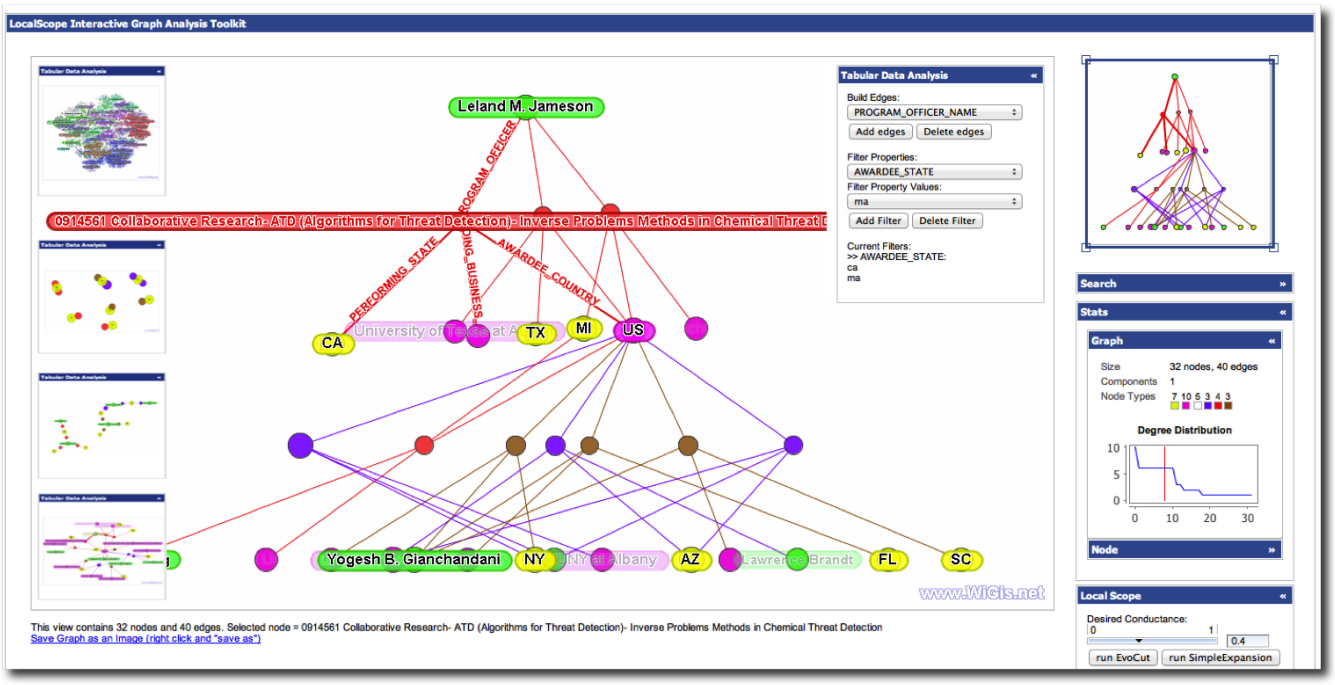


Figure 2. Example workflow in LinkScope, showing an interpolated layout of filtered NSF data.

Now that we have provided a high level overview of the LinkScope toolkit, we discuss is feature set more thoroughly. The next paragraphs organize the available tools based on our interaction types from Figure 1 earlier. Following this, a detailed description is provided for each graph interaction technique. LinkScope is implemented using a Java-based graph visualization framework called WiGis [10], which provides it with a standard set of interactions such as zoom, labeling, appearance adjustment etc. Further details of these are available in [10].

Graph Construction and Layout

Prior to visualization, the analyst can build a graph from one or more data tables by building edges from attribute types and/or filtering edge sets based on attribute values. Liu [15] describe a set of detailed mechanisms in for building graphs from tabular data, such as weighting, projection and aggregation, –all of which would be useful additions to our framework. In this paper we are interested primarily in mouse-based gesture mechanisms for graph interaction, and accordingly the discussion of tabular data ingestion here is relatively brief.

Edge Builder

LinkScope currently ingests tabular data by treating each row as an entity and each column header as an attribute for that entity. The initial graph contains a set of disconnected entities, each representing a row in the tabular data. The tab panel shown in Figure 2 allows an analyst to build a custom graph by adding attribute nodes relevant to her current task. Attribute nodes have labeled edges linking to entity nodes (NSF grants and Terrorism Incidents in the use cases presented later). Figure 2 shows an example of type labeling along attribute edges.

Edge Filter

Obviously the addition of attributes as graph nodes quickly increases graph connectivity leading to the classic "ball of string" problem. An edge filtering feature partially alleviates this by supporting visual examination of arbitrary subsets of nodes with particular attribute values. While there are other useful ways to perform this selection, for example, binning, pivoting are proximity grouping [15], the building and filtering process is sufficient to support our analysis of the usefulness of our interaction techniques. Furthermore, LinkScope supports a window-based filtering over ordered attributes such as time, as described next.

Time-based Slicing

Temporal distribution of events is critical for many analysis tasks. For example, [18] highlight the usefulness of temporal distribution by analyzing the hash tag #earthquake in twitter streams to pinpoint each aftershock in the 2010 Chilean earthquake. LinkScope supports visual slicing of ordered attributes such as time by running a layout on nodes within a specific time window. Analysts can click on a "play" button to step through the current window, incrementally adding nodes with the next time stamp to the visualization, based off a pre-computed layout for the entire time window.

Topic Modeling

LinkScope uses libraries from the TopicNets system [10] to perform LDA topic modeling on a graph prior to visualization in order to link nodes based on the topic similarity of their content. This feature allows the system to infer inter-node

similarity based on the topic associations, enabling it to work on free text documents with no structured meta data.

Automated Visualization Methods

Once an initial graph has been built, LinkScope can apply a range of "non-interactive" algorithms to produce various perspectives on the graph. Here, we define "non-interactive" as the set of algorithms where the analyst does not click on the graph itself. Most of the methods below need to be invoked via controls on the right panel, while the results are output directly on the node-link graph.

Content Highlighting

A simple, but very useful tool is the text-based search panel. Analysts can search over various abstractions of node contents, such as "Label", "Abstract" or "Full Contents" in our NSF scenario for instance. Search results appear in a list and can be further filtered via check-boxes in the search panel. Results are then highlighted on the graph in the main window.

Clustering

To provide an abstract view of the node-link topology of a graph, simple k-Means clustering is supported through a control on the right panel. The graph maintains a layered hierarchy so that an analyst can view graph connectivity at different levels of granularity using a slider. An additional feature to the clustering algorithm is that mouse interactions with a clustered, abstract view of a graph are mapped onto the full graph, and an analyst can view these in a window on the right panel.

Statistical View

Statistical views have proven to be a useful analytical aid for graph visualization [17, 12, 19], especially for larger, unfiltered and more highly connected graphs. LinkScope supports a set of three panels that provide simple statistical data insights about a currently selected subset of the graph. The first panel, shown on the right of Figure 2 provides basic stats at the global level. A degree distribution graph plots the number of nodes with a given degree, and overlays the currently selected node(s) as vertical red lines, indicating their position in the global distribution. A color-coded list shows the number of and size of disconnected components in the graph. This panel is interactive, and clicking on a distribution or component value will highlight the appropriate node in the main view. The second panel (not shown) computes pairwise statistics (such as shortest path for example) and is only displayed when exactly two nodes are selected. This supports automatic drawing of the shortest path between node selections on any visual configuration in the main window. Lastly, the node-level statistics panel (shown closed in Figure 2 provides detail about individual nodes/entities, for example, an analyst click though to the original entity/document that it represents. The panel shows a list of neighboring nodes, and supports drill-down for those also.

Layouts

The LinkScope analysis toolkit contains implementations of a number of force-directed graph layout algorithms including an optimized Fruchterman Reingold layout [10] Binary stress layout [14], FM3 layout [11] and the topic-based multidimensional scaling layout from [9]. The toolkit also uses layered layout based on Dk component-analysis [16] that attempts to lay out more connected components first.

INTERACTIVE GRAPH ANALYSIS

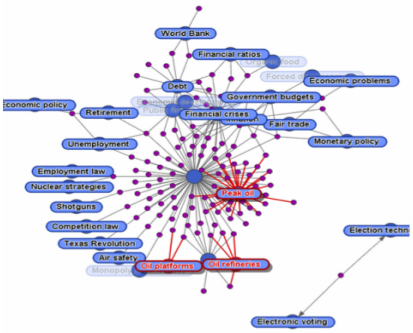
In this section, we describe our novel interaction algorithm in LinkScope, including a discussion of various design choices and implementation detail. The idea is based on our original interpolation method in [23]. The technique has been extended to support untangling highly connected graphs while preserving informative local structures. To clearly illustrate the effect produced with each technique, Figures 3 and 4 show the results on a simple bipartite graph. The nodes in the graph represent a set of news articles crawled from the New York Times website. A topic modeling algorithm [6] has been run over the contents of the articles, and additional nodes have been added to represent the resultant topics (i.e.: the term lists produced by the topic modeling algorithm). Edges are placed between documents and topic nodes if they have an association score above a threshold value.[9] provides further detail on generating document-topic graphs in this manner. Figure 3(a) shows an overview of this graph laid out with a simple force directed algorithm. Additionally, a text based search over the node contents has been performed for the keyword "oil" and the result nodes have been highlighted in red. The blue labeled nodes represent mined topics and the smaller red nodes represent individual news articles.

Interpolation method

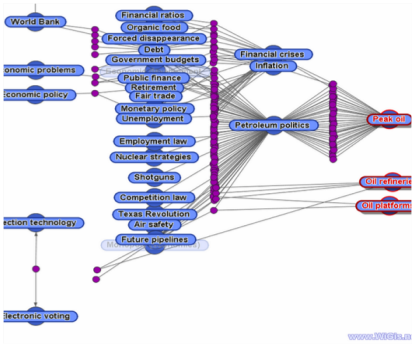
The original interpolation method is a simple approach to interactive graph layout. The algorithm is instantiated with one single node that the user clicks on. The displacement made on this target node is applied to all the other nodes with a weighted function that decays based on hop-distance from the moved node. This will deform the graph in a tree-like fashion. This method allows an analyst to "mold" a graph into different configurations which can a) better represent her mental model of the data, and b) create views that better communicate the connectivity of one or two target nodes. Figure 3(b) is reached from the state in Figure 3(a) in a simple mouse drag with our method. Now the graph is arranged in a tree-like fashion, pivoted around the three nodes containing the search keyword "oil";

Coupling Interpolation with Local Clustering Structure

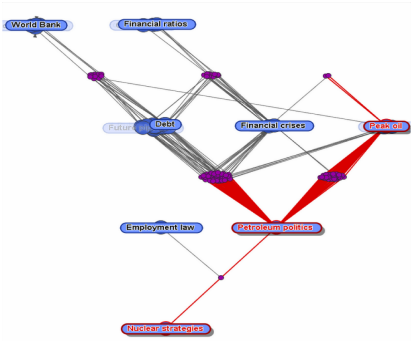
While the interpolation method can provide insight in some cases and is highly scalable, it only works well on trees and mesh-like graphs where a few click-and-drag would deform the graph in a natural looking way. For highly connected graphs with small diameters, the distance-based interpolation would generate less meaningful layout as it will destroy the nice local layout structures generated by static layout algorithms. To address this issue, we have combined the simple interpolation method with a number of different clustering approaches to allow selecting and dragging a entire cluster while maintaining the internal layout generated by the force directed layout algorithms.



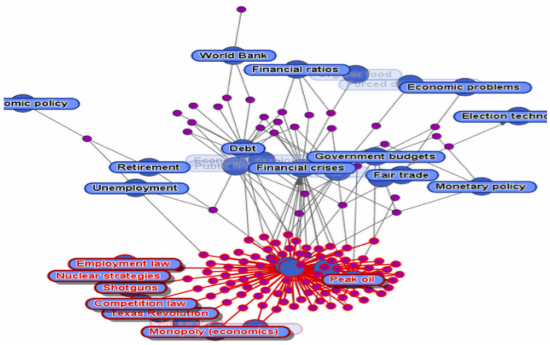
(a) A force directed layout of New York Times articles showing labeled topics. Articles are connected to topics if their LDA association is above a threshold.



(b) An interpolated layout based on results for the search query "oil". These nodes have been dragged to the right and are highlighted in red.



(c) An interpolated layout performing a pairwise comparison. From the previous view, the node for "Nuclear Strategies" has been dragged to the bottom of the screen to achieve this layout.



(d) An interpolated layout based on dragging a cluster of nodes selected from the seed node "Nuclear Strategies" using EvoCut clustering.

Figure 3. Example of interpolation-based layouts.

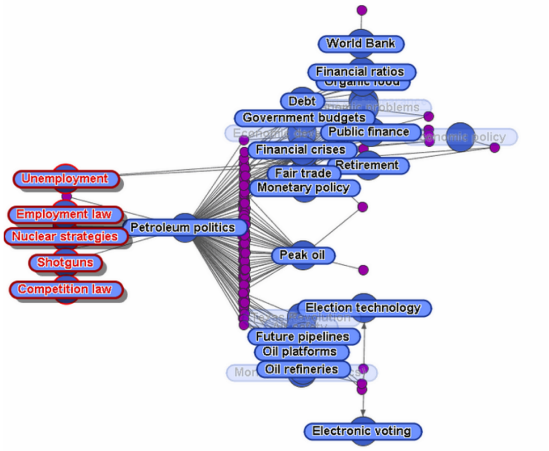


Figure 4. Example workflow in LinkScope, showing an interpolated layout of filtered NSF data.

The general idea is described as follows: the analyst select a local cluster based on certain criteria, the central node inside the cluster is used as the pivoting node and moved according to input mouse gesture. The weights of other nodes are computed using the interpolation method. However, for nodes inside the cluster, their weights will be constrained to 1 so they will moved exactly the same as the pivoting node. The weights of the rest of the nodes are decayed further so they would be moved further less to achieve better 'separation'. We also allow 'local refinement' by running different layout algorithms inside the cluster. The relative position of this cluster to the rest of the graph is registered. So if the analyst select another local cluster and choose 'Freeze Clustered Nodes', he/she can do the same interaction without affecting previous untangled structures. Thus the analyst can easily examine the relationship between different clusters and see how they are directly or indirectly linked together.

The key point here is how to effectively select meaningful local clusters. We have explored four possible ways and here we discuss the 'pros and cons' for each of them.

Simple neighborhood expansion

The first thing we tried is the neighborhood expansion based on hop distance. Usually a one-ring or two-rings neighborhood can gave a basic idea of what's going on around a node. However, sometimes a neighbor node is better off to be included in another cluster and it will be difficult to make this kind of decision without looking further away in the graph.

Geometric proximity based selection

The second variant on the interpolation technique focuses on geometric proximity as opposed to graph distance for selection of the cluster of nodes to be dragged. The idea behind this is that nodes which are similar to each other might not be directly connected. (e.g. in NSF data, nodes are indirectly connected based on similar attribute values). Force directed layout algorithms can do a good job of place these nodes in close proximity to each other. Figure 4 shows the result with a single drag from the original layout in our news article example. Figure 6 shows a second example: The graph shows

a filtered set of NSF grant with many different node clusters. The set is filtered to show only grants associated with CA and MA and two other states. The larger yellow nodes represent the states, and the smaller grant nodes are only connected to each other via state nodes. In such cases a topology-based clustering algorithm would fail to find the clusters that are visible in the layout, since there are no direct interconnecting edges between them. We employ a HashGrid method to isolate proximity clusters where edges are not necessarily present. For the NSF data graph, dragging the cluster that lies between CA and MA is analogous to executing a SQL query with a where clause on the STATE field. The advantage of the visualization is that an analyst can perform a relative analysis of many such queries at a quick glance.

'EvoCut' local clustering algorithm

To find a local cluster around a target node in a smarter way, we implemented the EvoCut local clustering algorithm designed by Andersen et al. [3]. A detailed description and complexity analysis can be fount at the original paper. The algorithm tries to find a local partitioning of the graph around a target node by simulating the volumed-biased evolving set process, which is a Markov chain on sets of vertices. It stops either when the desired conductance is achieved or the cost has exceeded a certain value. To our knowledge, EvoCut is by far the fastest local cut algorithm which offers a good balancing guarantee. However, as the process is based on a random work, the size of the cluster and the member nodes may vary significantly each time the analyst probes the same node.

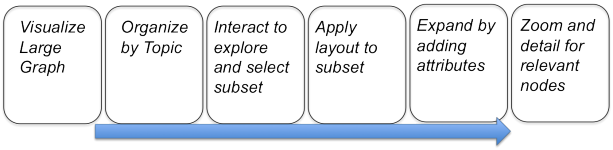
Content-based Selection

A logical application is to examine interpolation on node selections based on content. Figure 3(c) shows a further application of the method for the pairwise comparison of targeted nodes. In this case, the analyst is interested in learning about the topical relations between "Oil" and a one of the other labeled topics in the set: "Nuclear Strategies" (NS). The analyst clicks on the representative node in Figure 3(b) and drags it towards the bottom of the view, causing the graph to smoothly transition to the state in Figure 3(c) which is a clustered representation of all graph nodes based on their relative distance from nodes containing the keyword "oil" and the topic node representing "Nuclear Strategies". Interestingly, in this view, the topic node that lies directly between the 'oil' cluster and the "Nuclear Strategies" node is labeled "Petroleum Politics", which seems to make sense semantically. Note that topic labels were assigned manually by a third party expert based on a study of the term lists for each topic.

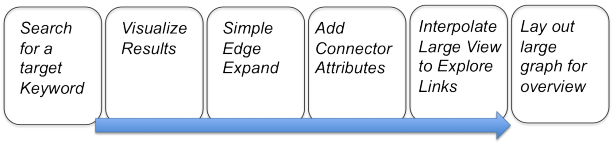
Example Workflows

The toolkit supports a diverse scope of visual analytics workflows. We now discuss a representative example for both top-down and bottom-up workflows in LinkScope.

An analyst can begin with a fine grained starting point such as a targeted search query, a particular individual, grant, topic or institution for example, and perform a variety of expansion functions over the network to arrive at a a broader informative visualization related to the initial seed. Figure 5(b) shows an example of a bottom-up workflow using the available tools. Importantly, LinkScope supports iteration over the steps in the workflow, enabling an analyst to handle complex tasks where necessary. A provenance view for workflows is also supported, as illustrated in Figure 2. Figure 5(a) shows an example of a top-down workflow where the analyst begins with a large graph visualization and invokes various filtering and search tools to arrive at content descriptions for relevant or interesting nodes.



(a) Example top-down workflow



(b) Example bottom-up workflow

Figure 5. Two example workflows in the LinkScope toolkit. In addition to the linear flow, workflows can be applied iteratively for more complicated analytical tasks.

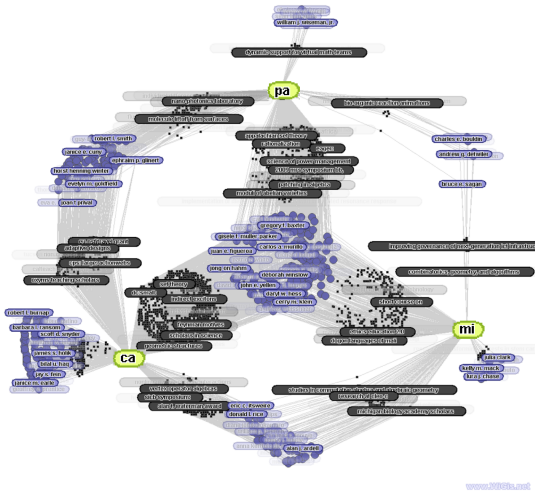


Figure 6. 3905 NSF awarded grants, with 6914 edge connections, for the states of PA, MI and CA from 2008 to 2010, showing STATE, TITLE and PROGRAM_MANAGER attributes. FM3 and Binary Stress layout models do not produce visually distinguishable clusters on this data.

CASE STUDY

In Section's and we have shown some of the interpolation methods working at a high level on on NSF data and on a corpus of New York Times news articles. Now we presnt a more detailed query driven use case based on analysis of a set of awarded NSF grants.

Analyzing NSF Grants

To illustrate some of the synergies between dynamic processing of structured data and interactive interpolation of resulting visual graphs in LinkScope, a corpus of 16,561 awarded NSF

grants were loaded into the system for analysis. We begin with a typical question that a high-ranking official might ask:

- *"For a given set of US states, who are the most active NSF program managers, and what grants did they fund?"*

The first step in our analysis is to apply a filter for grants that are connected with our target states. In this example, CA, PA and MI were chosen randomly. Following this, an edge builder was applied to produce a graph of all grants awarded in the target states. Next, the PROGRAM_MANAGER property was selected and representative nodes were added to the graph. Again, an edge builder linked them to the appropriate grant nodes. An edge filter ensured that only those PMs related to the selected states were included.

Several attempts were made to visualize the resulting graph of 3905 nodes and 6914 edges. FM3 and Binary Stress algorithms were used to lay out this data. Both algorithms produced cluttered looking layouts, with only a handful of clusters visible in the FM3 case. Next our interpolation method was applied to the graph. Three random mouse drags were used on state nodes to produce the layout shown in Figure 6. The resulting interpolated layout revealed a far more interesting structure than the other layout mechanisms. Clearly, the specific layout produced is influenced by the random mouse drags and this will produce inconsistent results. However, over many tials, the interpolation method consistently produced discernable (consistent node) clusters, whereas the other two algorithms did not.

In this example, program manager nodes are larger and blue, grants are black and state nodes are yellow and largest. A quick glance at the visualization provided enough insight to answer the probe question above. A proximity-based selection was made one of the discovered sub-clusters –in this case, PMs who awarded grants to CA and MI only, and an interpolation was performed over the cluster. Figure 7 shows a list of these PM names on the right side, and arranges the remainder of the graph (including the particular grants they awarded) relative to that list. Additional general information about distribution of grants by PMs across the three states also becomes clear from the visualization. For example:

- The PM who awarded the most grants was Jong On Ham, who awarded 493 in total, 46 of which went to CA, 2 to MI and 3 to PA. This information was highlighted by selecting the most connected components using a Dk technique [16] from the LinkScope statistics viewer.
- The total number of PMs across the three states was 443, and they awarded 3459 grants (gleaned from the statistics view panel on proximity-cluster selection). The majority went to CA (bottom left cluster in Figure 6 and less than a quarter of this amount went to MI (bottom right cluster).
- Most PMs awarded grants to all three states (central cluster), while very few awarded grants to just one state, with the possible exception of CA (bottom left cluster).

RELATED WORK

In this section, we discuss the state of the art in visual analytics. Our approaches are compared to others that attempt to combine interactive techniques with dynamic probing of additional attributes and values that can modify the structure and semantics of a graph.

A diverse collection of tools and applications for graph visualization and interaction are available either commercially or as open-source projects. For example, [2, 1, 20, 10, 7, 4] describe tools with a variety of approaches to visualization and feature analysis in graphs. LinkScope was designed to facilitate interactive graph modification though addition of metadata nodes from underlying tabular or relational data. Given that attribute values can be anything from simple text to nominal, ordinal or discretized numerical values, the ability to sort, organize and compare attribute values is an important design consideration. May [17] explore this issue in the context of generating an interactive visualization. Their SmartStripe method [17] allows a user to "step in to the feature subset selection process" to investigate relations and interdependencies between attributes and attribute values. Currently, LinkScope enables an analyst to perform such comparisons based on interpolated interactions with a node link graph structure. A logical next step for the system is to facilitate more detailed comparative analysis of attributes and values to allow for more fine grained querying. This challenge can be approached either directly, through informed graph manipulation with visual controls and queues, or in an associated tabular view that supports ranking, binning, pivoting and other comparative analyses for attribute values.

In addition to systems that support visual analysis over structured metadata, another relevant class of algorithms focuses on more ephemeral or probabilistic mechanisms for drawing inferences between data nodes in a graph. These methods are generally best suited for analysis of free text. Techniques such as latent semantic analysis, matrix decomposition and factorization methods such as Singular Value Decomposition (SVD), Principal Component Analysis (PCA), and neurocomputation methods such as self-organizing maps have been popular for producing visual representations of large bodies of text. For example, Luminoso [21] and GGobi [22] are text visualization systems that both employ SVD. Computations can be memory intensive for SVD and the resulting topics are not easily interpretable. Other multivariate analysis techniques are also popular in analysis of large text collections. For example Van Ham's PhraseNets [24] supports search for user-provided bi-grams (word pairs), which are then used to drive graph visualizations of large texts. Tools such as PhraseNets are exploratory and the result graphs produced (e.g.: a network of bi-grams) can be easily explored using interactive techniques such as LinkScope, most notably when a seed query returns a highly connected or very large graph. More recently, topic modeling [6, 5] has been used to infer associations between documents, and many tools have been developed [9, 8] to harness topic associations for visual analysis. For example, Gretarsson et al. describe an approach to searching a large corpus of documents based on topic association, where an interactive graph of documents and topics is visualized and refined through iterative application of an LDA [6] algorithm on a selected subset of visualized documents.

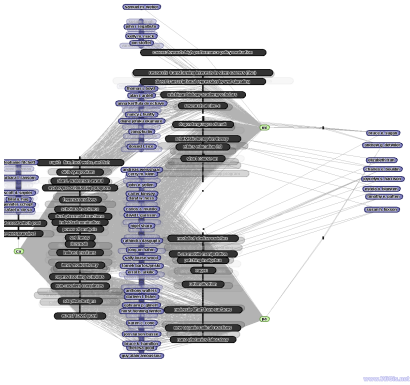


Figure 7. Interpolated view using proximity-based cluster selection. The view highlights a list of the program managers who awarded grants to both California and Michigan from 2008 to 2010. The remainder of the graph is structured relative to this node list.

DISCUSSION AND CONCLUSION

A key challenge in visual analytics is to deal with the uncertainty about properties such as size and connectivity of a result set as new search attributes and values are considered by the analyst. To address this challenge, we have introduced LinkScope, a toolkit for performing visual interactive analysis of structured or free text data, primarily though node-link data representations. The system supports three main tool types: 1) graph building tools and filters based on underlying tabular data, 2) visual components such as automated layout algorithms and parallel statistical representations, and 3) novel interactive tools for graph manipulation and exploration. Our research focused heavily on the interaction components, contributing several variants of a novel interpolation layout method based on mouse gestures. We believe that these method are useful for interactive analysis of node-link graphs, particularly when then can be coupled with dynamic attribute extraction from tabular data, and work best when attribute values are represented on the graph as connected nodes. The key innovation of the interpolation techniques is the application of methods traditionally reserved for automated graph layout and clustering to the task of interactive analysis. To evaluate our toolkit and particularly the interactive methods, we presented three analysis scenarios which required multiple graphs to built from tabular data, then analyzed interactively. These data included a corpus of New York Times text-only news articles, arranged by topic and a corpus of 16K awarded NSF grants between 2008 and 2010 with rich metadata. In each case our scenarios have shown how interactive interpolation methods can answer targeted analyst questions while revealing interesting related insights which may not have otherwise been considered by the analyst in the initial probe questions.

REFERENCES

1. Gephi : An open source software for exploring and manipulating networks, aug 2010.
2. Plotly: An interactive, collaborative tool for plotting streaming data online., jan 2015.
3. R. Andersen and Y. Peres. Finding sparse cuts locally using evolving sets. *Proceedings of the 41st annual ACM symposium on Theory of computing*, page 20, 2008.
4. D. Auber. Tulip-a huge graph visualization framework. *Graph Drawing Software*, pages 105–126, 2003.
5. D. M. Blei and J. D. Lafferty. Topic models. *Text Mining Theory and Applications*, 3(2):113–120, 2009.
6. D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
7. S. Bostandjiev, J. O'Donovan, C. Hall, B. Gretarsson, and T. Höllerer. Wigipedia: A tool for improving structured data in wikipedia. In *ICSC*, pages 328–335, 2011.
8. A. J. Cowell, M. L. Gregory, J. Bruce, J. Haack, D. Love, S. Rose, and A. H. Andrew. Understanding the dynamics of collaborative multi-party discourse. *Information Visualization*, 5(4):250–259, 2006.
9. J. O. Donovan, A. Asuncion, and D. Newman. Topicnets : Visual analysis of large text corpora with topic modeling. *ACM Transactions on*, V(5952):1–26, 2011.
10. B. Gretarsson, S. Bostandjiev, J. O'Donovan, and T. Höllerer. Wigis: A scalable framework for web-based interactive graph visualizations. In *GD'09: Proc. of the Intl Symposium on Graph Drawing*, 2009.
11. S. Hachul and M. Jnger. Large-graph layout with the fast multipole multilevel method. *Spring*, V(December):1–27, 2005.
12. J. Heer, S. K. Card, and J. A. Landay. *Prefuse: a toolkit for interactive information visualization*, volume 05pp, pages 421–430. ACM, 2005.
13. D. Keim, G. Andrienko, J.-D. Fekete, C. Grg, J. Kohlhammer, and G. Melanon. *Visual Analytics: Definition, Process, and Challenges*, volume 4950, pages 154–175. Springer, 2008.
14. Y. Koren and A. Civril. The binary stress model for graph drawing. *Graph Drawing*, 2(1):193–205, 2009.
15. Z. Liu. Network-based visual analysis of tabular data. *October*, 3(c):39–48, 2011.
16. P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat. Systematic topology analysis and generation using degree correlations. *Proceedings of the 2006 conference on Applications technologies architectures and protocols for computer communications SIGCOMM 06*, 36(4):135–146, 2006.
17. T. May, A. Bannach, J. Davey, and T. Ruppert. Guiding feature subset selection with an interactive visualization. *Most*, pages 109–118, 2011.
18. M. Mendoza, B. Poblete, and C. Castillo. *Twitter under crisis: can we trust what we RT?* ACM Press, 2010.
19. A. Perer and B. Shneiderman. Balancing systematic and flexible exploration of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):693–700, 2006.
20. M. Schaal, J. ODonovan, and B. Smyth. An analysis of topical proximity in the twitter social graph. In *Social Informatics*, pages 232–245. Springer Berlin Heidelberg, 2012.
21. R. Speer, C. Havasi, N. Treadway, and H. Lieberman. Finding your way in a multi-dimensional semantic space with luminoso. *Media*, pages 385–388, 2010.
22. D. F. Swayne, D. T. Lang, A. Buja, and D. Cook. Ggobi: evolving from xgobi into an extensible framework for interactive data visualization. *Computational Statistics & Data Analysis*, 43(4):423–444, 2003.
23. P. Trethewey and T. Höllerer. Interactive manipulation of large graph layouts. Tech report, Dept of Computer Science, UCSB, 2009.
24. F. Van Ham, M. Wattenberg, and F. B. Vigas. Mapping text with phrase nets. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1169–1176, 2009.