Maps of Computer Science

Daniel Fried*
Department of Computer Science
University of Arizona, Tucson, AZ, USA

Stephen G. Kobourov†
Department of Computer Science
University of Arizona, Tucson, AZ, USA

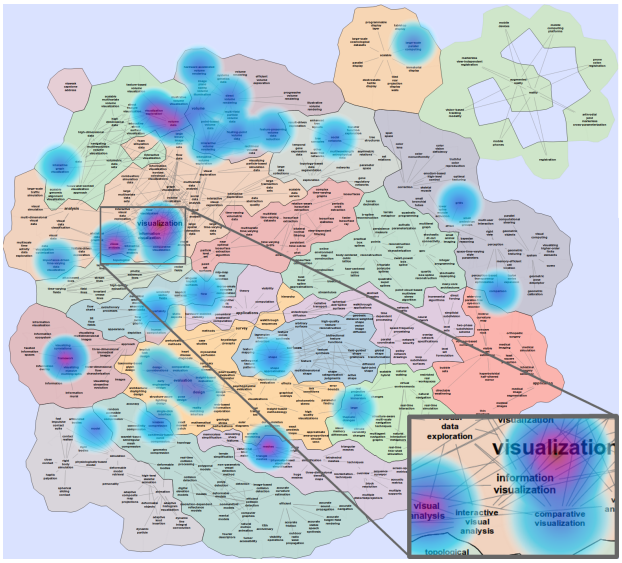


Figure 1: Map of TVCG based on 1,343 TVCG titles in DBLP, heatmap overlay based on 34 papers by the most prolific TVCG author. (Multi-Word Term extraction, C-Value with Unigrams ranking, Partial Match Jaccard Coefficient similarity, Pull Lesser Terms filtering, $N = 1500$.) The terms in the map are contained in 1,041 TVCG titles (78% coverage).

Abstract

We describe a practical approach for visual exploration of research papers. Specifically, we use the titles of papers from the DBLP database to create what we call *maps of computer science* (MoCS). Words and phrases from the paper titles are the cities in the map, and countries are created based on word and phrase similarity, calculated using co-occurence. With the help of heatmaps, we can visualize the *profile* of a particular conference or journal over the base map. Similarly, heatmap profiles can be made of individual researchers or groups such as a department. The visualization system also makes it possible to change the data used to generate the base map. For example, a specific journal or conference can be used to generate the base map and then the heatmap overlays can be used to show the evolution of research topics in the field over the years. As before, individual researchers or research group profiles can be visualized using heatmap overlays over a specific journal or conference base map. We outline a modular and extensible system for term extraction using natural language processing techniques, and show the applicability of methods of information retrieval to calculation of term similarity and creation of a topic map. The system is available at mocs.cs.arizona.edu.

Index Terms: H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—Clustering; H.3.1 [Information Storage and Retrieval]: Content Analysis and Indexing—Linguistic processing; H.5.m [Information Interfaces and Presentation]: Miscellaneous—Information visualization

*e-mail: dfried@cs.arizona.edu
†e-mail: kobourov@cs.arizona.edu

1 Introduction

Providing efficient and effective data visualization is a difficult challenge in many real-world software systems. One challenge lies in developing algorithmically efficient methods to visualize large and complex data sets. Another challenge is to develop effective visualizations that make the underlying patterns and trends easy to see. Even tougher is the challenge of providing interactive access, analysis, and filtering. All of these tasks become even more difficult with the size of the data sets in modern applications. In this paper we describe *maps of computer science* (MoCS), a functional visualization system for a large relational data set, based on spatialization and map representations.

Spatialization is the process of assigning 2D or 3D coordinates to abstract data points, ideally in such a way that the spatial mapping shares many characteristics with the original (higher dimensional) space. Multi-dimensional scaling (MDS), principal component analysis (PCA), and force-directed methods are among the standard techniques that allow us to spatialize high-dimensional data.

Map representations provide a way to visualize relational data with the help of conceptual maps as a data representation metaphor. Graphs are a standard way to visualize relational data, with the objects defining vertices and the relationships defining edges. It requires an additional step to get from graphs to maps: clusters of well-connected vertices form countries, and countries share borders when neighboring clusters are tightly interconnected.

In the process of data mining and data analysis, clustering is a very important step. Maps are helpful in visually representing clusters. First, by explicitly defining the boundary of the clusters and coloring the regions, we make the clustering information clear. Second, as most dimensionality-reduction techniques lead to a two-dimensional positioning of the data points, a map is a natural generalization. Finally, while it often takes us considerable effort to understand graphs, charts, and tables, a map representation is intuitive, as most people are

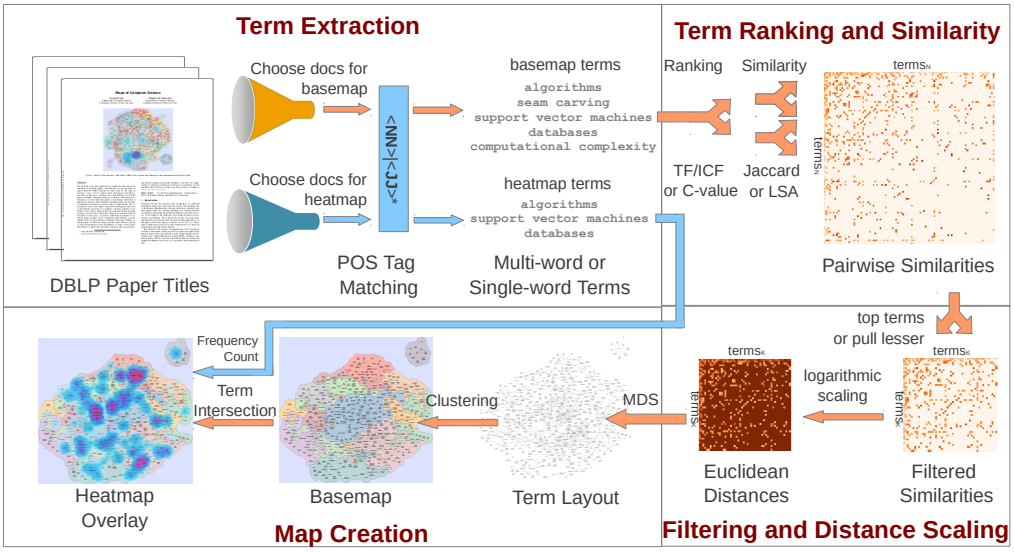


Figure 2: The main steps of the MoCS system are querying documents from DBLP, extracting terms from these titles, ranking terms by importance, calculating term similarity, further filtering terms based on similarity, and finally performing multidimensional scaling and clustering to produce a basemap, over which a heatmap can be overlaid.

familiar with maps and map-based interactions such as pan and zoom.

An overview of our MoCS system is in Figure 2 and our main contributions are as follows. First, we describe a visualization system which interactively generates *base maps of computer science* from the DBLP bibliography server [23]: from maps based on all 2,184,055 papers available in the database, to maps based on a particular journal or conference, to maps based on an individual researcher. These maps are generated from words and phrases extracted from the titles of the research papers in DBLP. Terms are selected based on their importance in the titles, and positioned and clustered according to co-occurrence similarities between terms. Some difficulties in making topic maps from research paper titles result from the short length of titles and the sparsity of terms, so we introduce several modifications designed to produce informative topic maps even from these short documents. Second, our system provides great flexibility in visualizing focus-data and context-data through a heatmap and basemap system. In particular, *temporal heatmap overlays* allow us to see the evolution of fields, journals, and conferences over time. *Individual heatmap overlays* allow the visualization of individual researchers in the field, or individual researchers in a particular conference, or individual papers in a particular conference. Finally, the MoCS system is modular, extensible, available online, and with complete source code, thus making it easy to change various components: from the natural language processing steps, to the creation of the graph that models the topics, to the visualization of the results.

2 Related Work

Using maps to visualize non-cartographic data has been considered in the context of spatialization by Skupin and Fabrikant [29] and Fabrikant *et al.* [13]. Rendering topical spaces as a map dates to work on term-maps by Callon *et al.* [8]. Work at PNNL resulted in document visualization systems such as Wise *et al.*'s Themescape [35], which used layers and terrain to represent text document corpora, and successive systems Spire and In-Spire. Recent systems include VOSviewer [30] and the Sci2 system [5], which provide an adaptable set of tools for spatial visualization of large document collections.

GMap uses the geographic map metaphor for visualizing relational data and was proposed in the context of visualizing recommendations, where the underlying data is TV shows and the similarity between them [16, 18]. This approach combines graph layout and graph clustering, together with appropriate coloring of the clusters and creating countries based on clusters and connectivity in the original graph. A comprehensive overview of graph based representations by von Landesberger *et al.* [33] considers visual graph representation, interaction, editing, and algorithmic analysis.

Word clouds and tag clouds have been in use for many years [28,31]. The popular tool, Wordle [32] took word clouds to the next level with high quality design, graphics, style and functionality. While these early approaches do not explicitly use semantic information such as word relatedness in placing the words in the cloud, several more recent approaches do. Koh *et al.* [21] use interaction to add semantic relationship in their ManiWordle approach. Parallel tag clouds by Collins *et al.* [9] are used to visualize evolution over time with the help of parallel coordinates. Cui *et al.* [10] couple trend charts with word clouds to keep semantic relationships, while visualizing evolution over time with help of force-directed methods. Wu *et al.* [36] introduce a method for creating semantic-preserving word clouds based on a seam-carving image processing method and an application of bubble sets. Paulovich *et al.* [27] combine semantic proximity with techniques for fitting word clouds inside general polygons.

There is a great deal of related work on natural language processing, text summarization, topic extraction and associated visualizations. Statistical topic modeling relies on machine learning techniques to extract semantic or thematic topics from a text collection, e.g., via Latent Semantic Analysis [11], or Latent Dirichlet Allocation [4]. Extensions to these topic models allow discovery of topics underlying multi-word phrases [34] and the use of additional syntactic structure, such as sentence parse trees, to aid inference of topics [7]. The topics provide an abstract representation of the text collection and are used for searching and categorization.

Motiviation for our work as a visualization of semantic topics in scientific research journals is summarized well by Mane and Börner [24], who introduce maps produced from scientific journals as a way to identify emerging research areas and the relationship between existing fields. Börner *et al.* [6] outline several of the techniques that we apply here, including co-occurrence similarity calculation and multi dimensional scaling.

Previous work has investigated visualization of search queries. Batagelj *et al.* [2] visualize coauthorship networks in DBLP corresponding to topical queries. WhatsOnWeb [17] visualizes web page search results as a graph hierarchically clustered by semantically-related pages. Document similarities are calculated

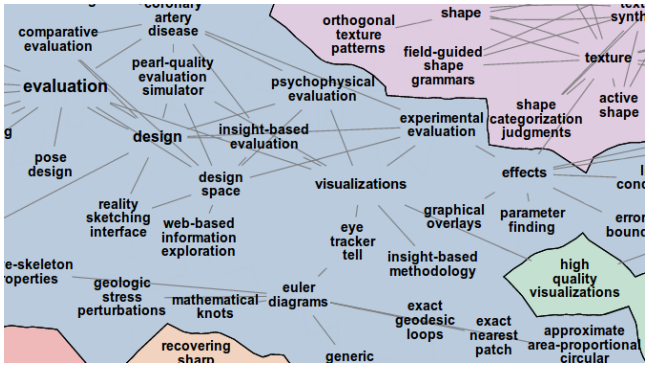


Figure 3: Section of a multi-word term map of 1,343 TVCG paper titles using the C-Value with Unigrams, Partial Match Jaccard Coefficient, and Pull Lesser Terms functions, with $N = 1500$.

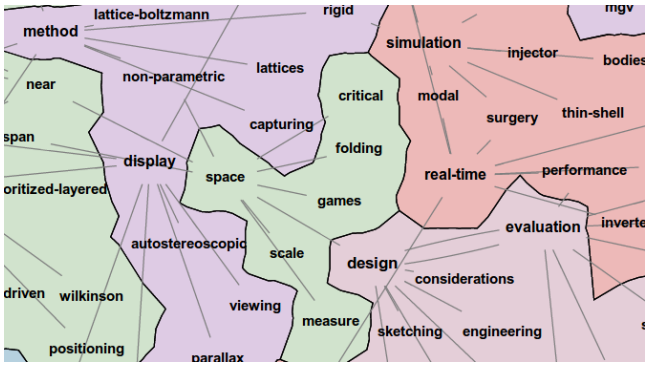


Figure 4: Section of a single-word term map of 1,343 TVCG paper titles using the TF, LSA, and Pull Lesser Terms functions, with $N = 1800$.

using common text between documents, weighted by term ranking. MoCS uses variations of these techniques to calculate term similarity using document co-ocurrence.

Our system extends the word- and topic-visualization systems referenced above with semantic, multi-word natural language processing techniques. Since multi-word terms can convey greater technical specificity than single-word terms, such multi-word extraction, ranking, and similarity algorithms are necessary to adequately visualize important research topics and their relationships to one another. Additionally, our system introduces adjustments to these algorithms designed to handle sparsity of data resulting from the multi-word nature of terms and the short length of documents (only paper titles, not abstracts or full text, are available in DBLP). Finally, the system allows visualization of a target set of documents within a context set of documents through the heatmap and basemap technique. Combined with the ability to sample and query from the titles of more than 2 million papers, these features constitute a system for visualizing specific thematic topics and their similarities in usage across a large number of documents.

3 Maps of Computer Science

Here we describe the main steps in the system: natural language processing (term extraction, term ranking, term filtering, similarity matrix), and graph and map generation (distance matrix, embedding, clustering, coloring).

3.1 Term Extraction

In the first step of map creation, multi-word terms are extracted from the titles of papers in DBLP. Part of speech (POS) tags are used to choose words that constitute topically meaningful terms, and exclude functional words (words that convey little semantic meaning, such as "the", "and", and "a"). The Natural Language Toolkit (NLTK) POS tagger [3] is used to label the words in all titles with POS tags. Once a title is tagged, maximal subsequences of words with POS tags matching the following regular expression are extracted from titles:

$$(\langle JJ\rangle|\langle JJR\rangle|\langle JJS\rangle|\langle NN\rangle|\langle NNS\rangle|\langle NNP\rangle|\langle NNPS\rangle)^*$$

JJ, *JJR*, and *JJS* are tags representing normal adjectives, comparative adjectives, and superlative adjectives, respectively, while *NN*, *NNS*, *NNP*, and *NNPS* are nouns, plural nouns, proper nouns, and proper plural nouns, respectively. This regular expression was chosen to extract a subset of noun and adjectival phrases including modifiers such as noun adjuncts and attributive adjectives.

Maps can be created with these multi-word terms (Figure 3), or the terms can be broken up into their constituent words (Figure 4) to parallel the word-based visual representations of systems such as Wordle [32].

3.2 Term Ranking

Once multi-word or single word terms are extracted, they can be assigned importance scores, or *weights*, based on their usage in the corpus of titles. Terms are then ordered by their weights to produce a ranking of terms, of which the top terms can be selected for inclusion in the visual map representation. We implement four such ranking functions in the MoCS system: Term Frequency, Term Frequency/Inverse Comparison Frequency, C-Value, and C-Value with Unigrams.

Under the *term frequency* ranking function, each term's weight is the number of times it occurred within the corpus. We seek to exclude common phrases that convey little semantic meaning, such as "introduction" (which occurs 9th in a list of multi-word terms ordered by frequency from a 1,000,000 title sample of DBLP, occurring 618 times). To accomplish this, term frequency–inverse document frequency (TF/IDF) [25] assigns a weight to a term proportional to its frequency in the document and inversely proportional to the number of other documents it appears in. In our domain, terms usually occur no more than once in each title. Therefore, we adapt TF/IDF to this task by treating the entire collection of titles as a single document, and counting the term's frequency in a reference corpus from a different domain to use as the inverse weighting value. We refer to the resulting method as *term frequency–inverse comparison frequency* (TF/ICF). The Brown Corpus [14], a selection of English text drawn from wide-distribution literature, is currently used as the comparison corpus.

C-value [15] is designed to account for possible nesting of multi-word terms (where short terms appear as word subsequences of longer terms). C-value incorporates total frequency of occurrence, frequency of occurrences of the term within other longer terms, the number of types of these longer terms, and the number of words in the term. The weight assigned by C-value is proportional to the logarithm of the number of words in a term, so we also include a modified implementation, *C-value With Unigrams*, that adds one to this length before taking the logarithm, allowing single-word terms to be assigned non-zero weight.

After terms are assigned importance weights, they are sorted in order of descending weight, and the top N terms are selected for possible inclusion in the map. N (Number of Terms) is a configurable parameter passed to the MoCS system. The value chosen for this parameter allows a large degree of control over the number of terms considered for inclusion in the map but greatly affects system runtime. We provide a default ($N = 1500$) that produces good maps relatively quickly for a wide range of map queries.

3.3 Similarity Matrix Computation

Once a set of top terms is selected, pairwise similarity values between top terms are calculated. We seek similarity functions that measure how closely the topics represented by two terms are related. Terms that refer to the same or similar topic, or topics that are closely associated, should receive high similarity values. We use

term-document co-occurrence as the basis of these similarity values, assuming that terms that appear together in multiple titles are more likely to be related in meaning.

The similarity functions take a term-document matrix, M, as input. The columns of M correspond to titles of papers from DBLP, and rows correspond to terms extracted by the term-extraction step. Each entry is the frequency of occurrences of the term indexed by the entry's row in the title indexed by the entry's column. We implement three similarity functions of this matrix M in the MoCS system: Latent Semantic Analysis [11], Jaccard Coefficient [19], and Partial Match Jaccard coefficient.

Latent Semantic Analysis (LSA) [11] is a method of extracting underlying semantic representation from the term-document matrix, M. A low-rank approximation to the term-document matrix is used to calculate the distance between terms in a vector-space representation reflecting meaning in topical space. The singular value decomposition of M is calculated using sparse-matrix methods, and rows in this decomposition represent terms as feature vectors in the high-dimensional semantic space. Terms are compared using cosine similarity [25] of the feature vectors to produce a matrix of pairwise similarities between terms.

Because of the short length of documents in our domain, the entries in the term-document matrix are effectively boolean. We provide *Jaccard coefficient* [19] as an alternative similarity function to accommodate the nearly boolean nature of the term-document matrix. Jaccard coefficient calculates pairwise term similarity as the number of documents two terms appeared together in, divided by the number of documents either term appeared in:

$$Jacc(S_i, S_j) = \frac{|S_i \cap S_j|}{|S_i \cup S_j|}$$

where S_i and S_j are the sets of documents containing the two terms. Jaccard coefficient alone treats terms as atomic units: multi-word terms only match if they are identical. This approach produces very sparse similarity matrices when used with a ranking algorithm such as C-value that prioritizes multi-word terms. To address this, we use a modification we refer to as *Partial Match Jaccard Coefficient*. This method attempts to address the sparsity of the C-value matrices by treating two terms as identical for the purpose of co-occurrence calculation if they contain a common subsequence of words.

3.4 Term Filtering and Distance Calculation

Term similarities have been calculated between the N highest ranked terms in the previous step. The next stage in the pipeline is *filtering*, choosing the terms to include in the map. We implement two filtering methods in the MoCS system: Top Terms and Pull Lesser Terms.

Top Terms is the simplest type of filtering, where we take the top-ranked K terms from the N highest ranked terms ($K \leq N$). The map is made from these terms and the terms' pairwise similarity values. Our current system has K set to 150. In practice, sparsity of data causes this method to produce fragmented maps, as the K terms often have low similarity to other top terms (particularly when the multi-word term-extraction system is used). We try to minimize fragmentation while still including a large number of terms in this map, and we chose this value of K to qualitatively balance fragmentation and map size after experimenting on a number of different document queries. However, Top Terms provides a decent default filtering method for single-word term maps, which have less of a long-tail frequency distribution than multi-word terms.

Pull Lesser Terms attempts to address the fragmentation of the top terms method, by using not only the highest ranked terms, but also including lesser-ranked terms if they are similar to a top-ranked term. Specifically, this method takes as input the N highest ranked terms, $terms_N$, and their pairwise similarities, as calculated in the ranking and similarity steps of the pipeline. The method plots the J highest ranked terms, $terms_J$, from among $terms_N$, and the $max\_associated\_terms$ number of most similar terms from $terms_N$ for each term in $terms_J$. Effectively, this method pulls in terms beyond the top J, if they are more similar to a top term than any of the other top terms. The default parameter values in our current system are $J = 90$, $max\_associated\_terms = 8$.

The pairwise term similarity matrix is next converted into a matrix of distances for use by the multi-dimensional scaling or force-directed algorithms of GMap. Let $S(t_i, t_j) \in [0,1]$ be the similarity between two terms, calculated using either LSA, Jaccard Coefficient, or Partial Match Jaccard Coefficient. Some choices of document sets and ranking and similarity functions produce terms with a similarity distribution more narrow than the theoretical range of the similarity function, so rescaled similarity values are calculated as

$$\hat{S}(t_i, t_j) = \frac{S(t_i, t_j)}{\max_{m,n:m \neq n} S(t_m, t_n)}.$$

The distance between these two terms, $D(t_i, t_j)$, is calculated using these rescaled similarity values as

$$D(t_1, t_2) = -\log[(1-\sigma) \cdot \hat{S}(t_1, t_2) + \sigma],$$

where σ is a small, positive, constant scaling value, currently set to 0.1, used to ensure a non-zero value inside the logarithm in the case that two terms have a pairwise similarity of 0. Linear transformations of similarities into distances produced layouts with dense term distributions and fragmented clusterings, which are less suitable for the map metaphor. A logarithmic scale allows comparison of relative distance between terms with low pairwise similarity by magnifying the distances between these terms.

3.5 Map Generation

We begin with a summary of the GMap algorithm for generating maps from static graphs [18]. The input to the algorithm is a set of terms and pairwise similarities between these terms, from which an undirected graph $G = (V, E)$ is extracted. The set of vertices V corresponds to the terms extracted from titles and the set of edges E corresponds to the top pairwise similarities between these terms as determined by the chosen filtering algorithm.

The number of edges created depends on the dataset but is limited by a parameter, max_edges, passed to the filtering procedure. For each term, the max_edges number of edges with the highest nonzero similarity values are included in the map. This prevents the graph from being overconstrained by very low term similarities, which would produce highly fragmented clusters. The final map includes the union of all these sets of highest ranked edges, so it is possible for a given term to have more than max_edges edges. We use a default value of 8 (chosen to correspond to the value of $max\_associated\_terms$ in Pull Lesser Terms) for this parameter.

In the first step of GMap the graph is embedded in the plane using a scalable force-directed algorithm [20] or multidimensional scaling (MDS) [22]. In the second step, a cluster analysis is performed in order to group vertices into clusters, using a modularity-based clustering algorithm [26].

We use information from the clustering to guide the MDS-based layout. In the third step of GMap, the geographic map corresponding to the data set is created, based on a modified Voronoi diagram of the vertices, which in turn is determined by the embedding and clustering. Here "countries" are created from clusters, and "continents" and "islands" are created from groups of neighboring countries. Borders between countries and at the periphery of continents and islands are intentionally modified, aiming for irregularity, which is typical of historical and geographic boundaries, and leads to more map-like results. Finally, colors are assigned with the goal that no two adjacent countries have colors that are too similar, using the SPECTRAL vertex labelling method [18].

To visualize the profile of a *target query* set of papers (for example, papers from a specified time range, author, conference, or journal)

over a map, we use heatmap overlays. Heatmaps highlight the terms in the basemap that also occur in the target query, with color intensity proportional to the frequency of the term's occurrence in the heatmap query. Separate database queries are used to produce the basemap and heatmaps (Figure 2), allowing a subset of the papers chosen for the basemap to be used for the heatmap. For example, a basemap can be constructed from a sample of all available papers, and a heatmap constructed from all papers for a particular journal (Figure 6), or a heatmap of a single author can be overlaid on a basemap of papers from a journal that author frequently publishes in (Figure 1).

The heatmap intensity, $I(t)$, is calculated for each term t that appears in both the heatmap and basemap query sets. These intensities are transformed on a logarithmic scale to allow terms with low I values to be visible in the heatmap, and then normalized so that the most frequently appearing term has intensity 1. The final normalized and rescaled intensity value, $I(t)$ is

$$I(t) = \frac{\log(F(t)+\beta)}{\max_{\hat{t}}[\log(F(\hat{t})+\beta)]}$$

where $F(t)$ is the frequency of the term in the hetamp query and β is a small additive constant (currently set to 1) that ensures terms that only appeared once in the heatmap query still receive a positive $I(t)$ value.

We plot these $I(t)$ values over terms using a blue and purple heatmap overlay. Each term t with $I(t) > 0$ has a semi-transparent circle laid over it, with the color intensity at the center of the circle scaling according to the value of the $I(t)$. In the current color scheme, this means that terms highlighed in purple were used more frequently than those highlighed in blue.Basemaps are rendered in the browser as vector graphics, and heatmaps are drawn as a semi-transparent raster overlay using the OpenLayers [1] heatmap implementation.

4 DBLP Visualization

A large map created from all 2,184,055 paper titles in DBLP is available online[1]. In this section we also provide examples of the ability of the heatmap and basemap system to visualize specific types of queries.

4.1 Individual Heatmap Overlays

The MoCS system allows separate database queries for the documents used to produce the basemap and the documents used to produce the heatmap overlay. Using the author information in DBLP, we can produce heatmap overlays of individual researchers over conferences and journals that they frequently publish in. Figure 5 shows a basemap constructed from titles of all papers published at the Conference on Neural Information Processing Systems (NIPS), with a heatmap constructed from the titles of papers by the most prolific author at NIPS. We see activity throughout the basemap, with particular intensity over a section of terms referring to inference in graphical models.

4.2 Conference and Journal Overlays

The bibliographic information stored in DBLP allows us to plot heatmaps of specific conferences and journals over a basemap of all documents. Figure 6 shows heatmaps of papers from four venues: the Computer Vision and Pattern Recognition conference (CVPR), the Symposium on Theory of Computing (STOC), the International Conference on Web Services (ICWS), and Transactions on Visualization and Computer Graphics (TVCG). These heatmaps are plotted from all available paper titles in the DBLP database for each venue. The basemap over which the heatmaps are plotted is made from 70,000 paper titles sampled uniformly from all entries in DBLP. Some similarities can be seen between the venues: all share relatively high intensity in their heatmaps over terms "application", "analysis", "method", and "evaluation" Some notable topical differences between venues also stand out. CVPR has a high intensity region in the northwest corner of the map over terms such as "images", "objects", and "recognition", while STOC has most high intensity in the northeast corner of the map, over terms related to "graphs" and "complexity". ICWS has a high intensity in the south of the map over terms "web services" and "systems" while TVCG is literally all over the map, as visualization is associated with all areas of computing: from visualization of algorithms to algorithms for visualization, from design and analysis to applications and systems.

[1] http://mocs.cs.arizona.edu/tiled/canonical/

4.3 Temporal Heatmap Overlays

Specifying different date ranges for heatmap queries allows the generation of maps that show how areas of research have spread across the topic basemaps over time. The maps in Figure 7 show how terms in the titles of papers published in the Journal of the ACM (JACM) have shifted over the past six decades, starting in 1954. The heatmap for papers from 1954-1963 has high intensity values over terms dealing with numerical and matrix methods. Computational complexity grows in intensity in the 1964-1973 map, and complexity and algorithmic bounds outpace numerical methods in 1974-1983. The algorithmic bound terms remain consistently intense throughout the remaining decades. A conspicuous trend is the narrowing focus of the journal over time: in the first four decades the topics are all over the map, but in the last decade the topics are concentrated around complexity, algorithms, and bounds.

5 Implementation

5.1 Modularity

The system is built with a modular design to accommodate future incorporation of additional algorithms for ranking, similarity, and filtering, as well as application to new document databases. We plan to expand the system's capabilities by testing the ability of other algorithms to produce maps that provide a better visual representation of the latent topic space. Source code for the system is available for others who wish to experiment with algorithms of their own.

5.2 Database

Paper titles and meta-information are stored in a SQL database, containing entries for 2,184,055 papers, journal articles, conference proceedings, theses, and books. This bibliographic information is parsed from an XML dump of DBLP entries, containing author, conference or journal, and date meta-information for each paper title [23].

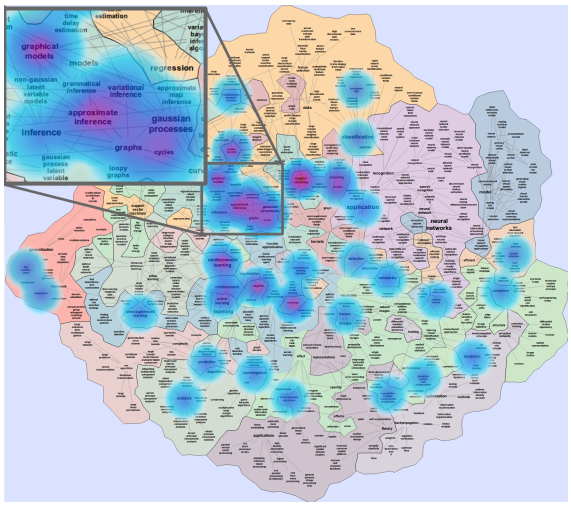
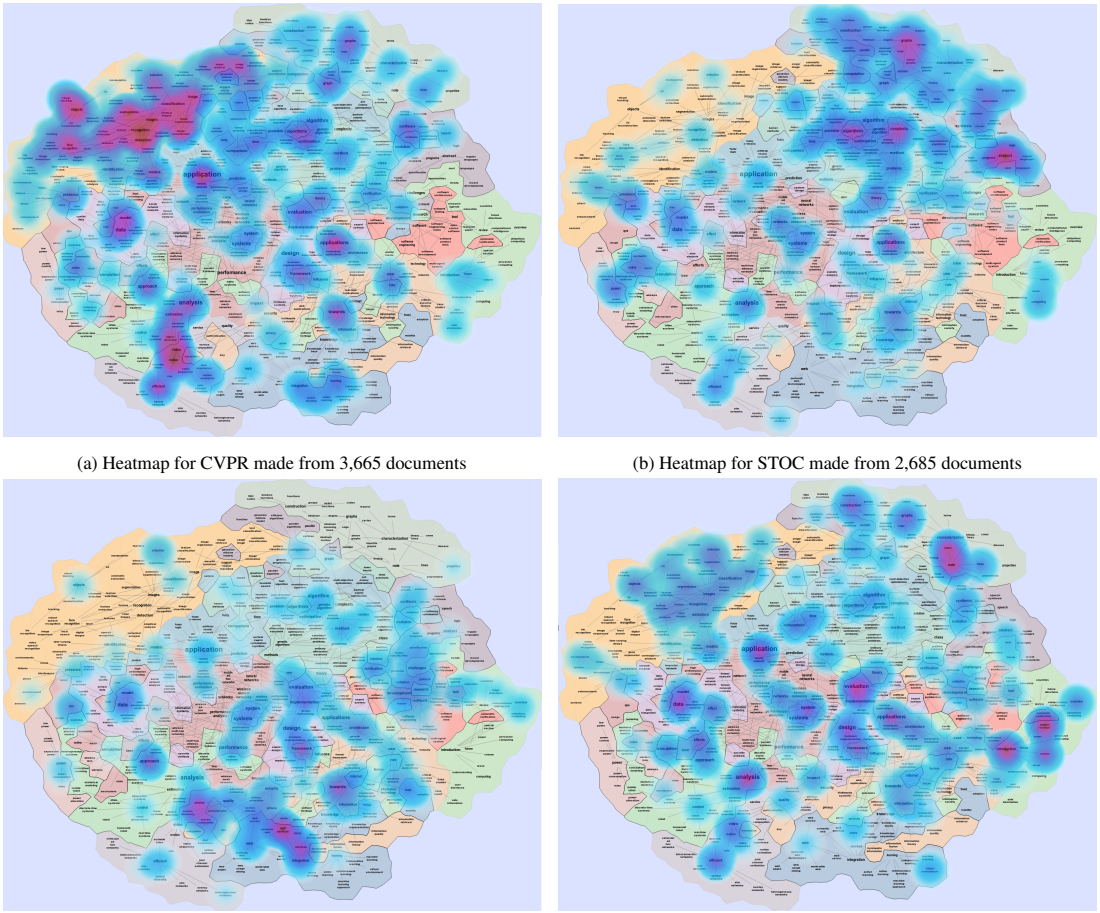


Figure 5: A heatmap produced from 75 papers by the author who has published most frequently at NIPS, over a basemap made from 3,553 NIPS papers, using C-Value with Unigrams ranking, Partial Match Jaccard Coefficient similarity, and Pull Lesser Terms filtering, with $N = 1100$. The basemap terms are contained in 2,770 NIPS documents (78% coverage).



(a) Heatmap for CVPR made from 3,665 documents

(b) Heatmap for STOC made from 2,685 documents

(c) Heatmap for ICWS made from 1,288 documents

(d) Heatmap for TVCG made from 1,826 documents

Figure 6: Conference and journal heatmaps overlaid on a map generated from 70,000 paper titles, sampled uniformly from all available DBLP papers, using C-value With Unigrams, Partial Match Jaccard Coefficient, and Pull Lesser Terms, with $N = 1500$. The basemap terms are contained in 1,660,311 of 2,184,055 DBLP papers (76% coverage).

Each paper is associated with its author and journal or conference if this information is available in DBLP. The database contains records for 1,324 journals, 6,904 conferences, and 1,237,445 authors which can be used to filter document title queries for map construction.

5.3 System Details and Runtime

The system is implemented using Python 2.7. NLTK [3] provides the POS labelling used for term extraction. The server is hosted in Django, using Celery as a back-end task manager, and SQLAlchemy for database interface. Maps are displayed in the user's browser using SVG rendering capabilities of AT&T's GraphViz system [12]. These SVG elements are rendered in a zoomable and pannable container provided by the open source OpenLayers JavaScript display library [1]. Heatmaps are overlaid with the heatmap plugin in OpenLayers and additional JavaScript for positioning the heatmap. Full source code is available at `mocs.cs.arizona.edu/code.php`.

MoCS currently runs in a virtual machine on a HP Proliant DL360 G6 server with 2 Intel Xeon CPU X5570@2.93GHz processors and 8GB of memory. Creating a map from 30,000 documents, sampled across all of DBLP, takes an average of 64 seconds with the default settings (multi-word phrases, $N = 1500$ terms, using C-value, Jaccard Partial Match, and Pull Lesser Terms). By stage, average runtimes are 30 seconds for sampling documents from the database, 1 second for ranking terms, 6 seconds for ordering terms and choosing the top, 25 seconds for similarity value calculation, 1 second for filtering, and 1 second for map drawing. Creating maps from a single author or conference is significantly faster, as the speed of the similarity calculation is heavily dependent on the number of documents sampled. Creating a map from all 2 million DBLP titles using C-value, Jaccard Partial Match, Pull Lesser Terms, and $N = 5000$ takes 8 hours.

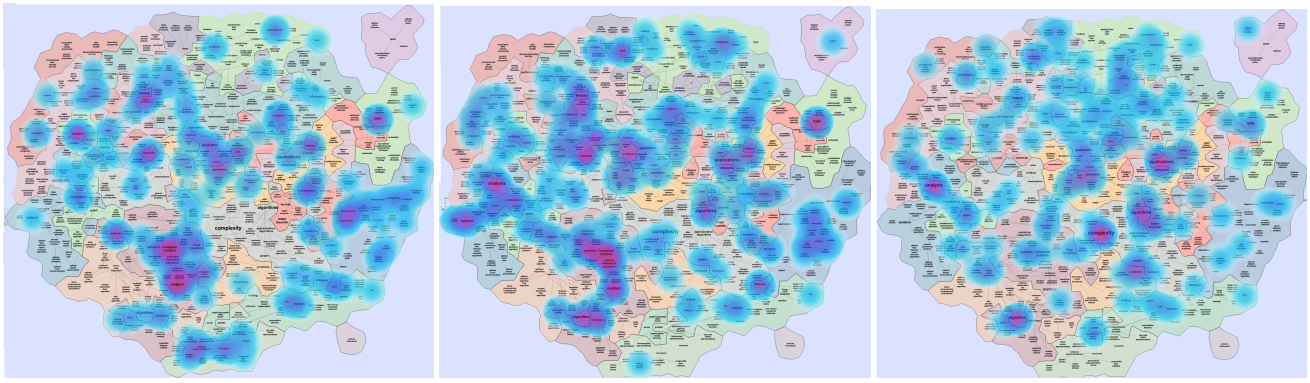
6 Discussion

Here we briefly summarize the novelty and utility of the MoCS system and discuss challenges in its implementation.

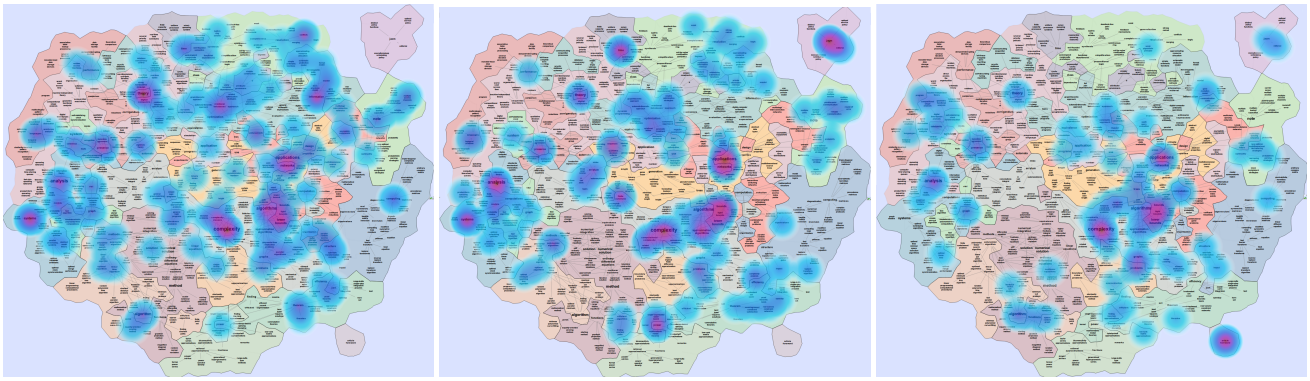
6.1 Novelty

While the MoCS system relies on standard NLP techniques to extract terms, rank terms and compute similarity between them, we had to solve non-trivial problems presented by the short documents and long-tailed frequency distribution of the terms in DBLP paper titles. In particular, the average title contains only 10.3 words and 2.9 multi-word terms. The multi-word terms have a long-tailed frequency distribution: there are over 2 million distinct terms in the database, but 80% of these occur only once. We have addressed these problems through application of two main strategies: exploiting the nested and overlapping nature of multi-word terms (through the Jaccard Partial Match and C-Value algorithms), and using similarities to inform inclusion in the map (in Pull Lesser Terms, a variant of query expansion).

While the use of spatialization and map-based metaphor for visualizing relational data is not new, our heatmap-basemap



(a) JACM heatmap for 1954-1963 (b) JACM heatmap for 1964-1973 (c) JACM heatmap for 1974-1983



(d) JACM heatmap for 1984-1993 (e) JACM heatmap for 1994-2003 (f) JACM heatmap for 2004-2013

Figure 7: Heatmaps of six decades of papers from Journal of the ACM (JACM). Basemap is generated from multi-word terms extracted from the titles of 1,998 paper titles published in JACM, using the C-Value with Unigrams ranking, Partial Match Jaccard Coefficient similarity, and Pull Lesser Terms filtering functions, with $N = 1400$. 200 paper titles were sampled uniformly from the JACM's publications for each decade to create the heatmaps. Full resolution images are available at `http://mocs.cs.arizona.edu/figures.php`. The basemap terms are contained in 1,543 titles from JACM (77% coverage).

visualization framework allows us a great deal of new flexibility in showing focus-data (e.g., research topics of a particular researcher) in the context of a larger set of the data (e.g., a conference that the researcher publishes frequently). Moreover, we can show pairs of focus-data/context-data at different levels of detail (e.g., research topics of a particular researcher in the context of the entire DBLP data), and over time (for example, how a researcher's interests have shifted over time by creating a heatmap of five years of his or her work over a basemap of all papers he or she has published).

Finally, our system is, to our knowledge, the only one of its kind that is fully functional online and provides complete source code. We have tried to design both the database backend and the ranking, similarity, and filtering stages in a modular fashion. This should allow extension of the system with additional algorithms for term ranking and similarity, and its eventual application to different databases of text documents such as grant proposals or collections of papers from other fields.

6.2 Utility

The MoCS framework would be useful in scientometrics, the science of measuring and analysing science research, and in the related fields of history of science and technology, and sociology of scientific knowledge. MoCS makes it possible to combine arbitrary basemaps with heatmap overlays, which allows great flexibility in document query visualization. For example, the work of a scientist can be viewed in the context of his or her venue of choice, a subfield, or the entire field of computer science. By filtering based on publication date, the system facilitates the identification of trends and patterns in research topics, e.g., the evolution of research topics over time for a given research venue, for a given researcher, or for an institution. A heatmap of a group of researchers over the map of CS can highlight the research strengths of an entire department, making it possible to summarize in a glance research across departments. This could be useful for prospective graduate students looking for departments with strengths in their area of interest.

Many journals (e.g., Cell, Earth and Planetary Science, Molecular Phylogenomics and Evolution) have recently added requirements for graphical abstracts as a part of research paper submissions. These are single-panel images designed to give readers an immediate understanding of the take-home message of the paper. MoCS can be used to generate graphical abstracts using a basemap from the journal and heatmap of the submission, positioning the paper on the venue's topic map.

6.3 Challenges

MoCS is our first prototype and has not been optimized for speed or memory usage. Scalability challenges can be overcome with better database management, optimization of the NLP and Map-generation algorithms, as well as precomputing similarities and query-specific filtering. As with most real systems, MoCS relies on several parameters. We have hard-wired a few (e.g., number of top terms, number of related terms, etc.) to values that seem to produce maps of manageable size. Several other parameters have default values but can be modified in advanced-query mode. Ideally, such parameters would be automatically set based on features of the specific query or resulting graph, but this is beyond the scope of this initial prototype.

The goal of good embedding of the graph in 2D space (using MDS) and the goal of clustering related nodes (using modularity) are often contradictory. This results in either fragmented maps (maps in which countries are made of many disconnected components)

or in maps where the countries are contiguous but at the expense of distorting the natural embedding. Implementing and testing different embedding/clustering algorithms and evaluating them for compatibility will likely alleviate some of these problems.

7 Conclusions and Future Work

In this paper we presented a practical approach for visualizing large-scale bibliographic data via natural language processing and using a geographic map metaphor. We described the `MoCS` system in the context of the DBLP bibliography server and demonstrated several possible exploratory visualization uses of the system.

We would have liked to compare the performance of our system against earlier and related approaches. However, this is nearly impossible as very few such systems are fully functional online or provide source code. We contacted the authors of several earlier semantic word-cloud or spatialization based systems but none were able to share source code or executables.

While ours is indeed a functional system, and it does offer various options for the natural language processing step, for the generation of the graph, and for the final map rendering, there are many possible future directions:

1. We can study departmental, state-wide, and even country-wide profiles over the base map of CS. This would hopefully allow us to visually compare and contrast the type of research done in different universities, states, and countries.
2. Automatic labeling of countries on the map can be accomplished by looking for the most frequent conferences and journals with topics in a particular country, and using the most relevant terms.
3. We plan to perform in-depth user studies to evaluate the effectiveness of the various algorithms. Do terms in the maps match what experts expect to see? Do similarities between terms reflect perceived semantic similarities between the represented topics?
4. The methodology described here is not limited to computer science research papers. We would like to generalize to other research areas, such as physics (ArXiv) and medicine (PubMed).

Acknowledgements

We thank Henry Kerschen for help with the the webpage for the *maps of computer science* server: `mocs.cs.arizona.edu`. We also thank Stephan Diehl, Sandiway Fong, Yifan Hu, and David Sidi for discussions about this project and ongoing system evaluation.

References

[1] OpenLayers: Free maps for the web. http://www.openlayers.org/.

[2] V. Batagelj, F.-J. Brandenburg, W. Didimo, G. Liotta, P. Palladino, and M. Patrignani. Visual analysis of large graphs using (X,Y)-clustering and hybrid visualizations. *IEEE Trans. Vis. Comput. Graph.*, 17(11):1587–1598, 2011.

[3] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python*. O'Reilly Media, Incorporated, 2009.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[5] K. Börner. Plug-and-play macroscopes. *Communications of the ACM*, 54(3):60–69, 2011.

[6] K. Börner, C. Chen, and K. Boyack. Visualizing knowledge domains. *Annual Review of Information Science and Technology*, 37(1):179–255, 2003.

[7] J. Boyd-Graber and D. M. Blei. Syntactic topic models. In *Neural Information Processing Systems*, 2008.

[8] M. Callon, J. Law, and A. Rip. *Mapping the Dynamics of Science and Technology: Sociology of Science in the Real World*. Macmillan Press, 1986.

[9] C. Collins, F. B. Viégas, and M. Wattenberg. Parallel tag clouds to explore and analyze faceted text corpora. In *IEEE VAST*, pages 91–98, 2009.

[10] W. Cui, Y. Wu, S. Liu, F. Wei, M. X. Zhou, and H. Qu. Context-preserving, dynamic word cloud visualization. *Computer Graphics and Applications*, 30:42–53, 2010.

[11] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[12] J. Ellson, E. R. Gansner, E. Koutsofios, S. C. North, and G. Woodhull. Graphviz - open source graph drawing tools. In *Graph Drawing*, pages 483–484, 2001.

[13] S. I. Fabrikant, D. R. Montello, and D. M. Mark. The distance-similarity metaphor in region-display spatializations. *IEEE Computer Graphics & Application*, 26:34–44, 2006.

[14] W. N. Francis and H. Kučera. *Manual of information to accompany a standard corpus of present-day edited American English, for use with digital computers*. Brown University, Department of Lingustics, 1979.

[15] K. Frantzi, S. Ananiadou, and H. Mima. Automatic recognition of multi-word terms: the C-value/NC-value method. *International Journal on Digital Libraries*, 3(2):115–130, 2000.

[16] E. Gansner, Y. Hu, S. Kobourov, and C. Volinsky. Putting recommendations on the map - visualizing clusters and relations. In *3rd ACM Conf. on Recommender Systems*, pages 345–348, 2009.

[17] E. D. Giacomo, W. Didimo, L. Grilli, and G. Liotta. Graph visualization techniques for web clustering engines. *IEEE Trans. Vis. Comput. Graph.*, 13(2):294–304, 2007.

[18] Y. Hu, E. Gansner, and S. Kobourov. Visualizing Graphs and Clusters as Maps. *IEEE Computer Graphics and Applications*, 99(1):54–66, 2010.

[19] P. Jaccard. *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz, 1901.

[20] S. G. Kobourov. Force-directed drawing algorithms. In R. Tamassia, editor, *Handbook of Graph Drawing and Visualization*, pages 383–408. CRC Press, 2013.

[21] K. Koh, B. Lee, B. H. Kim, and J. Seo. Maniwordle: Providing flexible control over wordle. *IEEE Trans. Vis. Comput. Graph.*, 16(6):1190–1197, 2010.

[22] J. B. Kruskal and M. Wish. *Multidimensional Scaling*. Sage Press, 1978.

[23] M. Ley. DBLP - some lessons learned. *PVLDB*, 2(2):1493–1500, 2009.

[24] K. Mane and K. Börner. Mapping topics and topic bursts in PNAS. *Proc. of the National Academy of Sciences*, 101:5287–5290, 2004.

[25] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge, 2008.

[26] M. E. J. Newman. Modularity and community structure in networks. *Proc. Natl. Acad. Sci. USA*, 103:8577–8582, 2006.

[27] F. V. Paulovich, F. M. B. Toledo, G. P. Telles, R. Minghim, and L. G. Nonato. Semantic wordification of document collections. *Computer Graphics Forum*, 31(3):1145–1153, 2012.

[28] A. W. Rivadeneira, D. M. Gruen, M. J. Muller, and D. R. Millen. Getting our head in the clouds: toward evaluation studies of tagclouds. In *CHI*, pages 995–998, 2007.

[29] A. Skupin and S. I. Fabrikant. Spatialization methods: a cartographic research agenda for non-geographic information visualization. *Cartography and Geographic Information Science*, 30:95–119, 2003.

[30] N. J. van Eck and L. Waltman. Software survey: VOSviewer, a computer program for bibliometric mapping. *Scientometrics*, 84(2):523–538, 2010.

[31] F. B. Viégas and M. Wattenberg. Timelines - tag clouds and the case for vernacular visualization. *Interactions*, 15(4):49–52, 2008.

[32] F. B. Viégas, M. Wattenberg, and J. Feinberg. Participatory visualization with wordle. *IEEE Trans. Vis. Comput. Graph.*, 15(6):1137–1144, 2009.

[33] T. von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. van Wijk, J.-D. Fekete, and D. Fellner. Visual analysis of large graphs: State-of-the-art and future research challenges. *Computer Graphics Forum*, 30(6):1719–1749, 2011.

[34] X. Wang, A. McCallum, and X. Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *IEEE International Conference on Data Mining (ICDM)*, pages 697–702, 2007.

[35] J. A. Wise, J. J. Thomas, K. Pennock, D. Lantrip, M. Pottier, A. Schur, and V. Crow. Visualizing the non-visual: spatial analysis and interaction with information from text documents. In *INFOVIS*, pages 51–58, 1995.

[36] Y. Wu, T. Provan, F. Wei, S. Liu, and K.-L. Ma. Semantic-preserving word clouds by seam carving. In *Computer Graphics Forum*, volume 30, pages 741–750, 2011.