PaperQuest: a Visualization Tool to Support Literature Review

Antoine Ponsard and Francisco Escalona

Abstract—The literature review is a key component of academic research, that allows researchers to build upon each other's work. Yet it requires researchers to browse hundreds of publications to find the ones related to their own work. While modern search engines enable fast access to publications, there is a lack of support for filtering out the vast majority of papers that are irrelevant to the current research focus. We present PaperQuest, a visualization tool that supports efficient decisions, by only displaying the information useful at a given step of the process. We propose a relevance algorithm to find and sort papers that are likely to be valuable to users, based on the papers they have already expressed interest in, as well as the number of citations. The current implementation uses papers from the CHI and UIST conferences, and citation counts from Google Scholar, but is easily extensible to other domains of the literature.

1 Introduction

The literature review is a key element of academic research. It allows researchers to build upon each other's work, and ensures that progress is being made. It is particularly critical for researchers entering a new domain, who are faced with the challenge of learning enough about it to make meaningful contributions. However, the scale of research in modern days makes this process difficult: even for a relatively well specified domain like Information Visualization, there are thousands of publications, spread into multiple conferences and journals, from authors scattered around the globe. Along with Zhang et al. [18], we believe that providing effective support for reviewing the literature can benefit the research community as a whole, and ultimately improve scientific productivity.

On any given topic, researchers must search and browse hundreds of publications to find which ones to read, and which ones to ignore. To avoid wasted time and effort, only publications that are meaningful and valuable to the researcher should be considered, so deciding what to read is a critical task. We discuss a multi-level decision process that accomplishes this as efficiently as possible, by gathering only the minimum amount of information on each paper to decide whether to keep it or not for the next level.

We present PaperQuest, an information visualization prototype that supports this multi-level decision process with simple interactions and an intuitive metaphor. A custom relevance algorithm finds and sorts papers that are likely to be valuable to users, based on the papers they have already expressed interest in. The number of citations is also taken into account, as a measure of the impact of a paper on the research field. Two additional views provide information on different facets of the data: the authors that frequently publish in this area, and the years of publication of the papers.

We begin, as customary, by reviewing the literature that addresses problems similar to ours. We then describe our data and task abstractions, present the general philosophy of our prototype, the underlying relevance algorithm that it uses, and discuss the visual encoding choices we made.

2 Related Work

Two high-level approaches to visualizing the scientific literature can be distinguished. Some emphasize its network aspect, and often use a node-link diagram representation; while others choose to display the multiple facets of the literature, such as authors, publication years, and keywords. However, relatively few solutions have been proposed to help users make sense of the information space around a particular topic — which is at the heart of the literature review process.

• *The authors are with the Computer Science Department of the University of British Columbia. Email: {aponsard, pax}@cs.ubc.ca.*

2.1 Literature as a Network

Many visualizations have been created to analyze the scientific literature, in particular in the Infovis community. Borner et al. use node-link diagrams to represent papers and authors, linked by citations or co-authoring relationships [1]. The size of the nodes encodes the number of citations received by a paper or an author, while the year of publication is encoded by the hue. The authors had to discard papers with less than 15 citations to avoid the hairball problem. CiteWiz provides several concept maps for influential authors and frequent keywords, but these lack readability because of the same hairball problem. It also features rather complex Growing Polygon techniques to visualize the influence of authors on each other [5].

To address the issue of the exponential number of links between papers, Citevis [13] proposes to show citation relationships by coloring nodes with the same hue, instead of drawing connection marks. The drawback of their interactive approach is that only a small subset of links are shown at a given time, which makes big picture views more difficult to obtain. Besides, only the most cited papers for each year were shown.

In contrast, the Citeology tool [11] displays three thousand CHI and UIST papers in a tiny font, stacked vertically for each year. "Parents" and "children" of a selected paper are highlighted and shown with smooth links. However, attempting to display the second or third generation usually results in an entangled set of links, especially for highly-cited papers. Keyvis [7] takes a different approach: after analyzing co-occurrence of keywords, it displays the results with large node-link diagrams and trees, to visualize trends in the Infovis literature.

2.2 Faceted Data Browsing

The hairball problem, and in general the large number of papers in the literature have encouraged alternative approaches to the node-link diagram. These approaches emphasize the faceted aspect of the metadata in the literature, and often use aggregation. Browsing faceted data has often been addressed, from the influential Flamenco system [17] to the recent idea of PivotPaths [4], in which selected metadata is used to transition continuously from one facet of the dataset to another, preserving the context.

In the context of the scientific literature, PaperLens [9]s aims at visualizing trends via a series of histograms that show the popularity of a keyword over time. It features multiple faceted views, such as one for selecting authors and one with a schematic representation of the ten most highly cited papers per year. These views are tightly linked, so that papers written by the selected authors are shown as colored layers in the topics histograms, for instance.

Netlens [8] is a general network visualizer that aggregates data for content and actors — in our case, papers and authors. The system is highly interactive: detailed views are shown for each type of data, and selection can be passed back and forth between the two facets of the data. While these approaches are interesting, they are not well suited

for a literature review, in which only a small subset of the literature is relevant to the user.

2.3 Literature Review as Sensemaking

Although many attempts have been made to visualize an entire domain of the literature, either with a node-link diagram or aggregate views, very few work actually addresses the task of a literature review. Instead of providing a top-down overview, there is a need to help users build a bottom-up understanding of a local neighborhood.

An early system was introduced by Mackinlay et al. [10], who represented each paper as a "butterfly", with its references on one wing and its citations on the other. At the time, the main concern of the authors was to handle very slow network connection to access remote databases. More recently, Zhang et al. proposed CiteSense [18], a text-based interface dedicated to searching, filtering and organizing papers during a literature review. After finding an interesting paper, the user can see papers that cite or are cited by it. The context of each citation is also provided, by showing the snippet of text in which the source paper is referenced. Some general-purpose sensemaking tools have also been used successfully for literature reviews, such as Jigsaw [14].

However, Apolo [3] is probably the closest to solving the problem we are addressing. From a seed paper, it fetches ten papers with a high number of citations, and tries to predict in which topic users are likely to consider them. They build upon the concept of the sensemaking loop [12] to explicitly support the creation and reorganization of an external mental representation of the domain of interest, subdivided into several user-defined topics. Our own system aims instead at finding the most relevant papers based on all those the user has expressed interest in, and provide easy access to the metadata and abstract of each paper.

3 Data

The scientific literature is an immense source of data, consisting of all the papers published, their metadata and relationships. Our own interests lie in the fields of HCI and InfoVis; for the sake of feasibility, we decided to focus on these two areas.

3.1 Datasets

Justin Matejka from Autodesk Research kindly agreed to share his own dataset with us, assembled for the Citeology tool [11], which contains papers and citations for the CHI and UIST conferences between 1982 and 2010. We also downloaded a dataset of InfoVis conference papers from 1995 to 2013 used for CiteVis [13], a project from the Il Lab at Georgia Tech.

Both datasets contain the paper title, DOI[1], year of publication and conference, authors, and references to other papers within the dataset. The Citeology dataset contains abstracts but no citation counts, while the CiteVis one includes citation counts but no abstracts. We extended the former by scraping Google for citation counts, and had originally planned to add abstracts to the CiteVis dataset. However, due to time limitations we did not manage to do this, and focused instead on the Citeology dataset. Covering the HCI field was a priority because we are more familiar with it, and so we could better evaluate the results of our relevance algorithm, which computes a derived attribute for each paper (cf. section Algorithm).

The references and citations form a directed network of papers. Another, independent network, arises from linking papers that have authors in common, and dually authors can be seen as nodes and papers as links between them. However, in the scope of this project we considered all data except references and citations as attributes of the papers themselves. Our data is therefore a multivariate network of 3501 nodes and 27587 directed links, of which 11697 are between papers inside the dataset.

[1] A digital object identifier (DOI) is a character string used to uniquely identify an electronic document.

3.2 Data Wrangling

The data had already been cleaned, so our efforts went towards completing the dataset. Google Scholar provides citation counts as part of its search results. It also protects itself against crawling and scraping, making it difficult to extract those data for large quantities of papers. We noticed that searching for the title of a paper in regular Google Search often includes the Scholar entry within the first 10 results, so we used the Kimono Labs tool[2], to get the first page from a Google Search of the title of each paper in our Citeology dataset. We then wrote a Python script to parse these results, filter them, and join them with our original data. The joining was done using the DOI, which works as a universally unique key for each paper. Unfortunately we could only find citation counts for 2848 (81%) of the papers with this technique, so further processing is necessary.

Finally, we traversed the network of references to compute the citations of each paper, and stored them in addition to its references. This redundant encoding does not increase dramatically the size of the dataset, and makes future computations much easier, by avoiding the need for a lookup in the entire dataset every time we need to access the citations of only one paper.

4 Task

It is difficult to begin a literature review effectively without an entry point in the relevant literature. Therefore, we assume that people have one or more *seed papers* available to them, usually provided by someone more knowledgeable about the field, or found by keyword search on Google Scholar. After *looking up* these seed papers, people want to discover related papers, which classically is done by browsing the references of the seed papers, a "backward search"; or the citations of these papers, a "forward search" nowadays available in many online libraries. The number of papers found in this process is potentially very large, because papers reference dozens of other papers and are sometimes cited hundreds of times. Reading each reference and citation is impossible.

Therefore, a crucial step in the literature review process is to *filter* the papers that have been found, to identify the ones that will provide the most information relevant to the domain of interest. An effective strategy is to conduct a *multi-level decision process*, in which one gathers on each paper only the minimal amount of information necessary to decide whether to keep this paper for the next level or not. In practice, we have observed that people seem to follow a similar approach:

1. Read paper titles, and keep only the ones that have a chance of being relevant. *One of our interviewees did so by opening each paper's ACM Digital Library page in a new tab of their web browser, by clicking on Google Scholar's search results.*

2. Read the metadata, the abstract and/or watch the accompanying video of the selected papers, to gather more detailed information on their content. *This is typically done on the publisher's website. However, some search engines such as Google Scholar or hcibib.org, embed a preview of this information in the search results themselves.*

3. Add the papers with the highest expected information gain to a "to read" list. *This can be accomplished by downloading the PDFs to a folder, or adding the papers to a reference manager.*

4. *Read papers from this list.*

5. Organize papers into different sub-categories, and annotate them.

This process is flexible and highly iterative: after reading some papers, one gets a better understanding of the domain of interest, and gathers new papers that will eventually be filtered and read. It is also common to do small-scale iterations, such as going back to reading paper titles after reading the abstracts of a few papers ($2 \rightarrow 1$). The multi-level decision process affords some sort of *batch processing*,

[2] https://www.kimonolabs.com/

where there is often more than one paper being considered at each step. While this is not required, we believe that the cognitive cost of task switching repels users from processing only one paper at a time. Figure 1 presents a summary of this process, and the appropriate task abstractions.

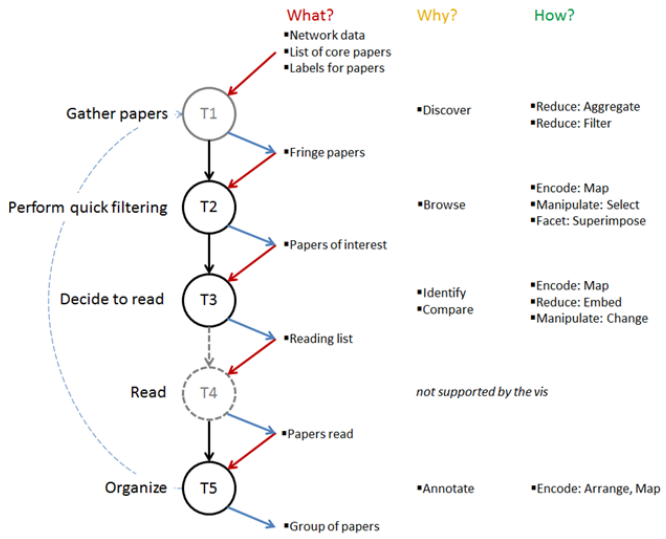

Fig. 1. Task description. T1: the algorithm aggregates papers metadata and relationships to compute a relevance score, and filters papers in the Fringe based on this score. T2: encode citation count with size, and superimpose links between papers on demand. T3: embed abstract and metadata in the paper list view, and let users reorder it. T5: classify papers into different subtopics. The dashed blue arrow shows that the literature review is iterative; we could have added similar arrows from any task to any previous task.

Finally, an important aspect underlined by Chau et al. [3] is that people build a mental representation of the domain they are exploring by classifying papers into different sub-topics. This classification can happen during any of the steps described above, as soon as enough information has been collected on the paper. Yet it may change significantly towards the end of the literature review, when a better mental representation has been found. The resulting classification is commonly used to write subsections of the "Related Work" section of a paper.

5 Design Concept

A literature review is an exploration of the space of previously published papers. This space can be divided into three subspaces, which we identify as:

- The Core: papers you have read, upon which you build your understanding of the field;
- The Fringe: papers you have access to, because they reference or are cited by some papers from the Core;
- The Unknown, an immense and terrifying abyss made of all the papers away from the Core.

As you make progress in the literature review, some papers from the Fringe will be added to the Core, which in turn will cause new papers to enter the Fringe. However, most of the papers will remain forever in the Unknown. To support the multi-stage filtering process described earlier, we add another subspace: the To Read list, consisting of the papers that you have picked from the Fringe, but have not read yet. This temporary buffer space is made necessary by the fact that reading papers usually takes much longer than the other steps of the filtering process, such as reading paper titles.

In the citations network, we define the Fringe as the papers that are one hop away from a paper in the Core or the To Read list. The problem of exponential explosion described in the Tasks section applies here: each paper references and can be cited by many other papers. Yet, most of these papers are probably irrelevant to your particular domain of interest. We therefore propose to order the Fringe based on a relevance score computed for each paper, which takes into account the number of citation links between this paper and those you have expressed interest in, as well as the number of citations of this paper. In the next section, we address the problem of normalizing these different pieces of information; and in the following one we present an algorithm for computing and combining them into a single relevance score.

6 Normalization

To determine the relevance of a paper we use three quantitative metrics: the internal citation count, computed from the dataset; the external citation count, scraped from Google; and a connectedness measure, computed by our algorithm (see details in the next section). However, these three metrics have very different scales. We normalize them to $[0, 1]$ to allow meaningful comparisons, both in the algorithm and the visual encoding.

Histograms of the internal and external citation counts show that they follow a similar power law distribution (Figure 2), with a majority of papers with zero or a few citations, and a long tail of papers with ultimately very high citations counts. The scale of these distributions is however drastically different, with a ratio of approximately 20. The scatterplot in Figure 3 shows that there is no strong correlation between internal and external citation counts: for instance, a paper cited only once internally can have anything from a handful to several hundred external citations. Since most of the papers are clustered in the bottom left of the scatterplot, we define two normalization parameters that encompass most papers: 20 for internal citations, 400 for external citations. The citation counts in each category are then divided by the appropriate normalization parameter so that they vary in approximately the same range. We enforce a strict maximum of one, to keep all values in the $[0, 1]$ interval. This means that the two normalization parameters defined above act as *cutoff points*, above which papers are simply considered to have the maximum possible citation count.

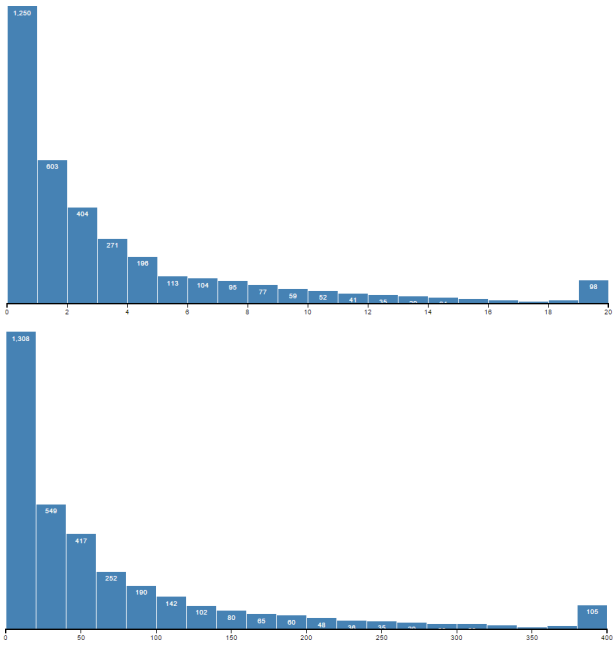

Fig. 2. (top) Distribution of the internal citation counts. (bottom) Distribution of the external citation counts, scraped from Google.

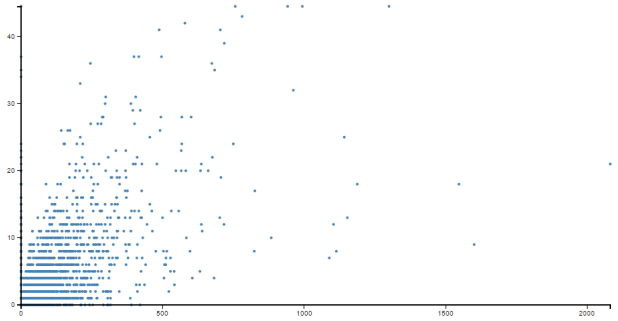

Fig. 3. Internal versus external citation counts. The vertical axis has been clamped halfway to show more detailed patterns at the bottom.

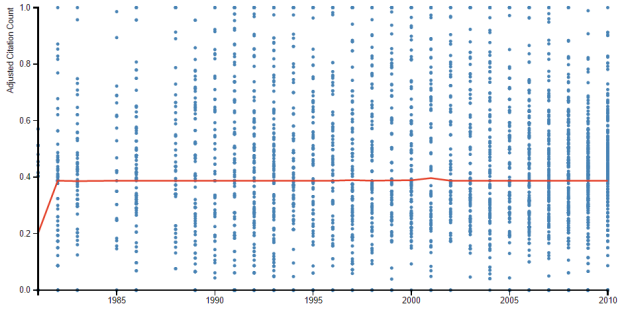

Fig. 5. Adjusted Citation Count of papers published in a given year. The red line is the median of the median maximum normalized citation count for each year.

Finally, we apply a non-linear transformation to the normalized counts, in order to spread apart the values at the bottom of the $[0, 1]$ interval. Indeed, a paper cited five times is often much better than one that has never been cited, whereas there is not much difference between papers cited 95 and 100 times. We used the square root function because it maps $[0, 1]$ to itself and has a vertical tangent in 0, where we want the maximum discernibility.

In addition to normalizing between internal and external citations, we have to make sure there is no imbalance between years of publication. It seemed plausible that old papers would be cited more often that newer ones, which simply did not have time to reach high citation counts. Figure 4 (a,b) shows that this is not the case, at least for the CHI and UIST conferences. Recency seems to affect papers only for the first five or six years, and old papers are not cited more often. It seems in fact to be the opposite: the median citation count is increasing over time.

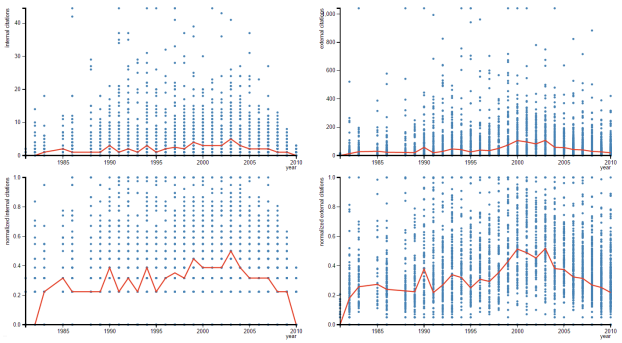

Fig. 4. (a) top left: internal citation counts of papers published in a given year. (b) top right: external. (c) bottom left: normalized internal citation counts. (d) bottom right: normalized external. We use medians as a measure of central tendency to reduce the influence of outliers.

For our algorithm, we combine the internal and external citation counts by taking the maximum of the *normalized* citation counts (MNCC). This means that we are considering a paper as important if it is well cited either in its own domain, or highly cited in other domains. Besides, it accounts for the fact that many papers were missing an external citation count, due to the imperfections of our scraping method.

The variations of the normalized internal and external CC over time are even stronger than before normalization (Figure 4 c,d). We therefore decided to also normalize across years, so that the median normalized citation count is the same in each year. We achieved this by computing the median of the median MNCC for each year, then linearly interpolate papers in each year to bring the median to the desired value (Figure 5). The resulting twice-normalized citation count is used as an input to our relevance algorithm, and referred to as the Adjusted Citation Count (ACC).

All of the graphs presented in this section are computed on-the-fly for the papers dataset, and accessible on the "dataset statistics" page. However, they are not intended for casual users, but only for advanced users who want to understand better their data and, eventually, set their own normalization parameters.

The normalization of the connectedness measure is more straightforward: we simply compute the minimum and maximum connectedness measures for papers in the Fringe, and linearly transform all connectedness measures to $[0, 1]$.

7 Relevance Algorithm

The purpose of the relevance algorithm is to find papers related to those that the user found interesting, and to compute how strongly related they are. Relatedness is, however, hard to define, and even harder to compute. Our dataset does not contain authors' keywords, nor any kind of hierarchical organization. Natural Language Processing techniques might be able to identify clusters of papers, but were beyond the scope of this project. Instead, we rely on one fundamental characteristic of the scientific literature: the fact that authors cite previous work that they build upon. By interpreting these citations as links in a network, we define relatedness as connectedness.

7.1 Connectedness Measure

The algorithm works on multiple sets of papers. The *Interesting* papers are the ones for which the user has expressed some interest, either by adding them to the *Core*, to the *To Read* list, or by *selecting* them on the Fringe. The *Fringe* is the one-hop neighborhood of the interesting set: papers that either cite or are referenced by at least one interesting paper. Note that the set of *Selected* papers represents the intersection of the Fringe and the interesting papers. Their union is the set of *Known* papers; all the other papers are considered *Unknown*.

For each Known paper, we compute a *connectedness measure* as the weighted sum of all the links between this paper and other papers in the Interesting set. The weights represent the level of interest of the user for each paper contributing to the connectedness measure. We infer this interest from the set these papers belong to: 1 for Selected papers, 3 for To Read, and 5 for Core. We do not make any distinction between references and citations, as we consider both to be indicative of relatedness.

The connectedness measure must be recomputed every time a paper P is moved from one interesting set to another, because its weight changes. However, this change affects only the papers that reference or are cited by P. For computational efficiency, the connectedness measure is computed incrementally: when P moves to another set, its contribution to the connectedness measure of its neighbors is incremented by the difference between its previous weight and its new one.

7.2 Relevance Score

The relevance score of a paper is computed as the sum of its normalized connectedness and its Adjusted Citation Count. We chose addition over multiplication because we consider high connectedness and

high citation counts to be enough on their own to make a paper relevant. Indeed, a paper not well cited, but strongly connected to other interesting papers could provide you pertinent insights, even though this paper may not be useful to the research community at large. Similarly, it is good to be aware of highly cited papers in your field, even if they are only loosely connected to your current focus.

The relative weight of the normalized connectedness versus the Adjusted Citation Count is another free parameter in our relevance algorithm. After testing our system on a set of papers related to our research interests, we decided to keep this relative weight to one, as we did not find any reason to favor one above the other. A better approach would be to deploy our system with real users, and collect which papers they selected. We could then train a machine learning algorithm on this data to find the parameters that lead to the best prediction, hence bring interesting papers higher up on the Fringe.

7.3 Delayed Execution

Papers on the Fringe are sorted vertically by relevance score. Every time the user selects a paper, or moves it to the To Read list or the Core, the relevance scores are updated, and the order of papers on the Fringe should change. However, these continuous updates may be distracting for users, especially if they are trying to read the Fringe from top to bottom and selecting papers from it. Therefore we provide a mechanism to delay the reordering of the Fringe by adding an "update Fringe" button. Users can still opt-in for continuous updates by ticking a checkbox. To support the delayed execution of the relevance algorithm, we maintain a queue of all the actions taken by the user since the previous update, and empty this queue when the update button is pressed.

8 Visualization

The relevance algorithm is a key component of PaperQuest, and we believe that the main benefit of our tool is its ability to find and sort papers based on all the papers you consider relevant, and not only one — as is today the case with backward and forward search. Therefore our visualization is organized around the suggestions of the algorithm.

8.1 Main View

A key decision of our design is to display paper titles in full, and to show many of them on the Fringe. Reading titles is the first step of the decision process, and the most efficient for filtering out the vast majority of irrelevant papers. The imperfections of the relevance algorithm make it necessary to display multiple titles at once, to let the user skim quickly through a list of suggested paper titles. To this end, we use spatial position to order papers in the Fringe, with the most relevant items appearing at the top (Figure 6). This top-down ordering corresponds to the reading direction in many cultures, which facilitates skimming and intuitively conveys which papers are the most relevant.

Our visual layout is therefore quite different from the force-directed node-link diagram commonly used to represent network data, such as the one used by Borner et al. [1]. Node-link diagrams are very effective for understanding the topology of a network, but our task analysis suggests that this is not particularly useful in the context of a literature review. That is why we do not display links between papers by default: the user has to request them by clicking on a button. Instead, we provide higher-level information on how connected a paper is to the set of interesting papers, as explained below. Besides, a force-directed node-link diagram does not afford a particular reading order, so it could have been difficult for users to know where to start, or to keep track of their progress — especially if the layout is being updated as new papers come in and out.

The main view contains two other regions: the Core and the To Read list (Figure 6). Users can move papers between these regions by clicking the appropriate buttons in the contextual menu that appears when hovering on a paper. The boundaries between regions can be dragged, which allow users to give more screen real estate to the region of their current focus. Papers are also aligned vertically in the Core and the To Read list, to make it easier to draw links between them and papers on the Fringe, as well as showing the full titles when these views are expanded. The slight curvature of the Fringe is intended to reinforce the conceptual design of a core surrounded by a fringe, so that papers seem to move from the *outside* to the *inside* when they are moved from right to left in the interface. The concavity could also help reduce the number of overlaps when drawing a link between two papers on the Fringe, although our current drawing technique does not leverage this fully.

Users can select and deselect papers on the fringe simply by clicking on their titles. The selected papers appear in bold, and are moved slightly to the left (Figure 6). Because we use the left click for selection, other actions must be performed via a contextual menu, such as adding a paper to the To Read list. The reason why we gave priority to selection is that it is the basis for two important features of our visualization: interactive fringe exploration and semantic zoom.

Selecting a paper on the Fringe signals to the algorithm that the user may be interested in that paper, and might be interested in its neighbors — based on our connectedness assumption. We offer the user the possibility to update the Fringe automatically, taking into account this new information to improve the prediction. The exact interaction is as follows: when the user presses the mouse button on a paper, it moves to the left and is shown as selected; when the user releases the mouse button, the Fringe is updated, which often causes the newly selected paper to move somewhere else in the Fringe. We implemented this subtle two-step action to give maximum control to the user, and avoid them being overwhelmed by multiple changes happening in the interface simultaneously. For the same reason, we always stagger animations: papers move first to their new position, then change color if needed.

When users select papers, we consider that they have a higher degree of interest in them. We apply the concept of Generalized Fisheye introduced by Furnas [6], to show more information on the selected papers than the non-selected ones. In our case, we show the list of the authors, the conference and the publication year below the title (Figure 7a). Users can even choose to focus entirely on the selected papers, and see their full abstracts, while the non-selected papers are shrunk and disappear (Figure 7b). This sequence can be seen as a semantic zoom, during which more and more information is revealed about the selected papers. We chose to map the transition between the different levels of zoom to the mouse wheel, which is often used for this purpose. Since we do not limit the number of papers that users can select, it is often the case that their full abstracts will not fit on one page. Therefore, any further scrolling action at the maximal zoom level will be interpreted as actual scrolling, and shifting the papers upward. The user can scroll in the opposite direction to move back to the top of the list, and keep scrolling to unzoom.

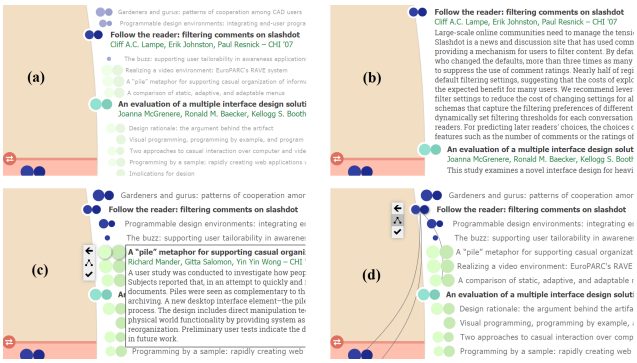

Fig. 7. Detail of some of PaperQuest's functionality. a) In the first level of semantic zoom, selected papers are highlighted and their metadata shown, while the rest of the papers are shrunk and faded. b) At the next level, the full abstract of selected papers is shown, and other papers are hidden. c) Ctrl-click can be used to see the details on a paper on demand in any view. d) The mouse hovers over a paper, highlighting it and showing its menu; the "show links" option is enabled.

The combination of selection in one click and semantic zoom by

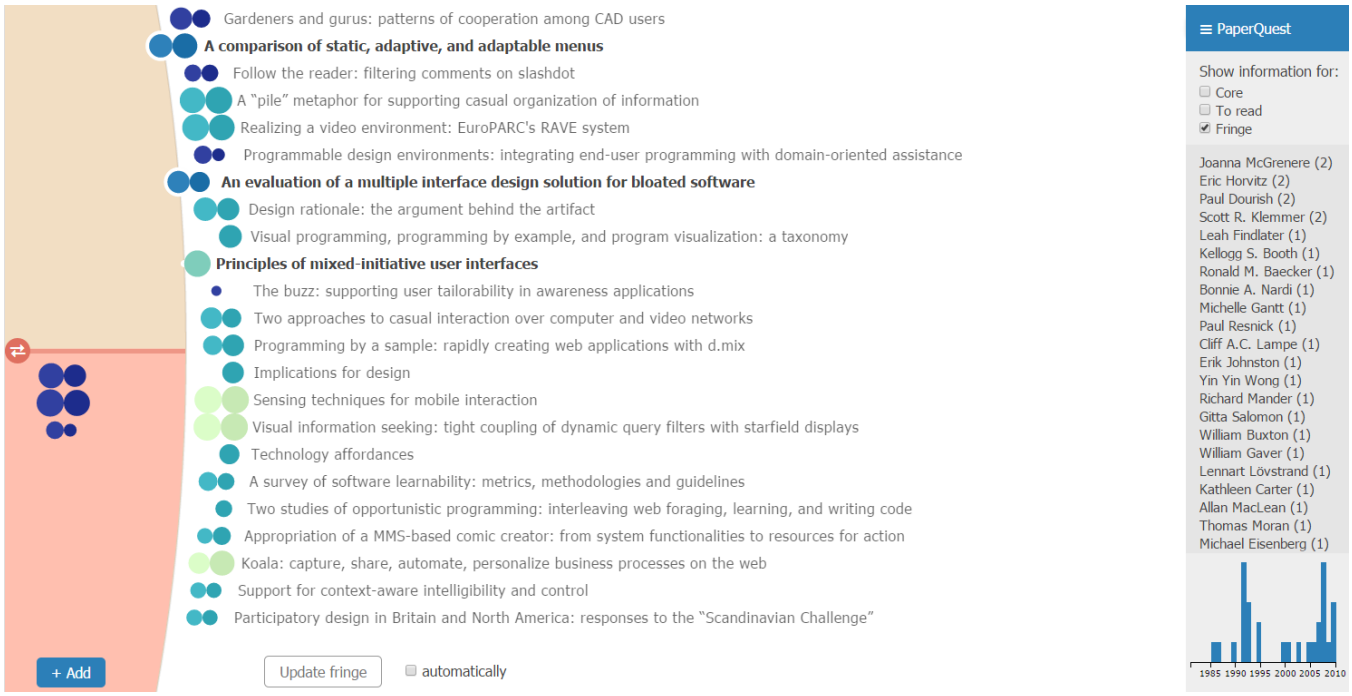

Fig. 6. The main interface of Paper Quest. The list of papers in the middle is the Fringe, at the top left is the To Read list, and at the bottom left is the Core. The right pane is the Sidebar. Below the Fringe an update button is faded out, indicating the Fringe is up to date. Three papers have been selected in the Fringe and three more are already present in the Core. The border of the Fringe and the border between the Core and the To Read list can be dragged to adjust the views.

scrolling makes it very fast to access the metadata and the abstract of papers of interest to the user. We also provide details-on-demand for individual papers, by displaying their information in an overlay when the user presses Ctrl and clicks on a paper title (Figure 7c). Finally, users can open the ACM digital library entry for a given paper by clicking on the list of authors, which is shown as a hyperlink. Users can thus access the PDF of the paper, and any supplemental material available.

In some cases, it is useful to know which paper cites or is cited by others. Users can display all the links of a paper by clicking on the appropriate button in the contextual menu (Figure 7d). We show links as curved arcs as suggested by van den Elzen and van Wijk [15], with the clockwise curvature indicating that the source makes reference to the target. References and citations are easily distinguishable without clutter from extra markers such as arrowheads. Only one paper at a time can show its links, and only to papers that are visible in one of the views.

8.2 Glyph Design

The algorithm combines internal and external citation counts, as well as a custom connectedness measure, into a single score. Not matter how advanced this algorithm might be, it will always make wrong predictions, or simply will not match the criteria of the user at a particular time. For this reason, we decouple the three quantitative pieces of information available for each paper, and display them as a glyph to the left of the paper title.

Because internal and external citation counts are semantically similar, we encode them in the same way: as the area of a disk. It is somewhat less effective than using, for instance, the length of a rectangle, which allows more accurate comparisons. However, our task analysis suggests that exact judgments are not required in this context: users simply need to quickly get a sense of how popular a give paper is. Besides, representing papers with a disk is reminiscent of node-link diagrams, and conveys the idea of treating the literature as a network of papers. The area of the circles in the glyphs are not normalized across publication years, but only between internal and external citations. This way, the visual encoding provides slightly different information than the vertical sorting by the algorithm, which gives the user a richer picture of the data. The exact citation counts, as well as the normalized ones, are available on demand in a tooltip that appears when hovering on the glyph.

We chose to encode the connectedness measure with luminance, because this channel is nicely separable from the size channel. As explained above, we consider connectedness and citation counts as two orthogonal metrics, but equally relevant. However, the number of distinguishable luminance levels seemed too small for our purposes: only two to four, according to Ware [16]. We expected that four or five distinguishable bins would be useful, to separate the top papers from the regular ones, and those from the only weakly connected ones. We also had to make sure that our glyphs would be visible on the white background, which of course restricts the extent of the luminance scale available. For these reasons, we decided to map the connectedness measure to a sequential color scale with monotonically increasing luminance, which we retrieved and adjusted from ColorBrewer[3].

Our glyph is therefore made of two colored disks. We draw them tangent to each other to form a single visual entity that encodes the relevance of a particular paper (Figure 8a). The external citations disk is always on the left side, which is clearly distinguishable from the internal citations disk on the right side, and allows comparisons with other papers on the Fringe. This distinction is effective, but not particularly intuitive. We added a very light shading to the internal citation disk to convey the idea of centrality. This luminance difference was chosen to be too small to interfere with any of the other bins in the color scale, but enough to be perceptible in a side-by-side comparison in the glyph itself.

We experimented with a variant of this glyph, in which internal and external citations are represented by adjoined half disks, instead of tangent full disks (Figure 8b). This "half-moons" variant is more concise than the "butterfly" one, which in turn may allow more accurate comparisons. After presenting the variants to several potential users,

[3] http://www.colorbrewer2.org/, by Cynthia Brewer and Mark Harrower

Fig. 8. Two versions of the paper glyph to encode internal and external citation counts. a) The "butterfly" variant consists of two disks side by side. b) The "half-moons" are two half disks. The external citation count is always displayed on the left.

we did not find any reason to favor one against the other, so we kept both. An entry in the top-right menu let users specify their personal preference.

8.3 Additional Views

We provide two linked views in a sidebar (Figure 6), to display other facets of the information shown in the main view. First, a list of the most frequent authors is shown, with the number of papers they co-authored in parentheses. This list tells the user which authors are particularly active in the domain they are exploring. Only a few authors appear more than once; but the list of authors who appear in exactly one paper is too long to be displayed in its entirety. We sort with several criteria. According to the idea of a generalized fisheye described earlier, we promote authors of the papers that users have expressed interest in by selecting them. We then give a higher weight to authors that have published a paper with a high relevance score. Among the authors of a given paper, we promote those that are more often first or last authors.

A histogram at the bottom right corner of the screen shows aggregate information on the publication years of the papers appearing in the main view (Figure 6). Its purpose is two-fold: to provide a sense of the popularity over time of the topic the user is currently exploring; and to identify potential gaps in their set of Core and To Read papers, compared to those that appear on the Fringe. Such a mismatch would indicate that the user is not aware of a related subtrend in their domain. Three checkboxes at the top of the sidebar let users define the set of papers they want to see information on. By ticking multiple checkboxes, users can aggregate multiple sets of papers, such as the Core and the To Read list.

9 Implementation

One of our goals was to make a tool that could be useful in practice. Creating a flexible system with simple but powerful interactions plays a part in reaching this goal. We chose the web as a platform to reach many people easily, and Data-Driven Documents, or D3 [2], to help us implement our design, with the rest of the functionality hand-crafted in JavaScript. We believe this combination of technologies enabled us to get very close to our original vision, with HTML and CSS providing a scaffolding for the application and an easy way to tweak the visual design of some elements. We also use the typeahead.js library[4] from Twitter for the auto-complete enabled search box, which in turn uses jQuery[5].

9.1 Code Structure

When the application is run, a simple HTML structure is put in place just so that the JavaScript libraries can run and take over the rendering of the interface. After the libraries, the full dataset is loaded. A session manager implemented on the browser's Local Storage recovers any state that was stored previously, if any. Only dynamic state is stored, no paper metadata or derived attributes that can be recomputed are saved. It also loads a global dictionary of configuration parameters for the visualization, which determine things like the time taken for the animations, good default values for the different view dimensions, etc.

The dataset is fed to the application as a JSON object, a dictionary with papers indexed by DOI, where each paper is itself a dictionary containing the original metadata: title, year, list of authors, abstract, list of citations and list of references. We also compute some derived attributes for each paper, including its relevance score, and flags indicating where in the three views it is located. To better encapsulate both the static and dynamic data, we created a *paper object* which provides utility functions, such as geometry helpers to determine the X and Y position of a paper based on its current state.

Geometry computations happen take into account the current state of papers and views, semantic zoom level, as well as configuration parameters. They are used frequently as the interface gets updated and re-painted, enabling a dynamic and responsive visualization that adapts to events like changes in window size.

We also created a global object called P to standardize access to the dataset. It can be used as a function to lookup a paper by DOI, returning all its data and state. P can be queried to get the current lists of papers in each view. It is effectively an abstraction layer that hides the details of how the data is handled and cached from the higher level code that implements the functionality. Its implementation relies on an array that contains the papers the user has showed interest in, either because they are selected in the fringe, or located in the To Read or Core views. It also contains all papers one hop away from these in the graph, which make up the non-selected fringe.

To establish the visible fringe, which consists of the elements of the fringe that can be seen in the interface, first the relevance score of each paper is updated using the algorithm described in Section 7, then the papers are sorted by this value and the top ones are marked as visible, as long as they fit on the screen. The papers are shown as a list with a slight curvature that conforms to the curvature of the fringe view, which is based on a circle with a large radius.

As the user interacts with the visualization its state changes. For example, papers in the fringe can be selected, and there are three references to papers in special states: one paper at a time can show its links, including references and citations; one paper can be actively highlighted by hovering over it, which also displays a menu; one paper can be expanded to see its metadata and abstract, and this information is actively shown until the user chooses to hide it. This state is stored in a global dictionary which is also persisted when the user saves their session.

The stats page was implemented separately, and while it uses the same data, it also derives new values to display bar charts, scatterplots and line plots, all of which are rendered using D3 directly.

9.2 Implementation Details

Most of the functionality for the different components of the interface was implemented by hand, with two exceptions. First, the transitions that happen when we add or remove papers and links are handled neatly by D3, as well as all the animations. Second, the textbox that lets users search for and add new papers by name is implemented using typeahead.js from Twitter.

The views with draggable edges, glyphs, menus, sidebar, stats page, and paper details overlay are relatively straightforward. Their implementation is based on listening to events in the browser and updating the application state accordingly. A single drawing function takes care of rendering everything according to the current state.

Semantic zoom in the Fringe is implemented by having a global zoom level parameter that gets updated when the mouse wheel is scrolled. Each discrete event of the wheel changes the zoom level to one of three values: titles only, titles with metadata, and full details including abstract. The last two expand papers that are selected, while using a transformation matrix to shrink the other ones.

The curved links are implemented using a spline with basis interpolation between three consecutive points: source, midpoint and target. Source and target are given by the coordinates of respective paper glyphs. The midpoint is computed by finding the middle point of the segment defined by source and target and offsetting it orthogonally. The offset amount is a function of a scaling parameter between 0 and 1, and the length of the segment, so that longer links will have a bigger offset than shorter links.

[4] https://twitter.github.io/typeahead.js/
[5] http://jquery.com/

10 Results

Figure 6 shows the main screen of our system. As mentioned above, we assumed that users already possess one or a few seed papers beforehand, which are added to the Core by default. The following scenarios illustrates two aspects of the workflow in PaperQuest. The first scenario presents how users can move through the interface during the different steps of their literature review; the second highlights how the visualization encodings help users make better choices.

10.1 Scenario 1

Panagiota is a PhD student in the field of HCI. She has just finished reading a couple papers for her literature review and now needs more material. To find new papers, she goes to the PaperQuest website, which shows her an overview of where she was at the end of her previous session (Figure 9). She can quickly see previous papers she has read, the ones in her reading list, and an outdated list of papers in her fringe of research. She marks as read the papers she has just finished reading (Figure 10a), which sends them into the core region at the bottom left of the interface. She then clicks the "Update fringe" button to get fresh recommendations of papers in her Fringe, and clicks on a few with promising titles to mark them as selected (Figure 10b).

Using the mouse wheel she zooms into the fringe to see more detail. The interface shows the metadata and abstract of her selected papers (Figure 10c). After reading their abstracts she deselects a couple of papers from the list, zooms out, updates the fringe again and repeats the process a few times. Once she is satisfied with her choices, she adds her selected papers to the To Read list (Figure 10d). She then brews a cup of coffee, gathers her courage, and gets back to her readings.

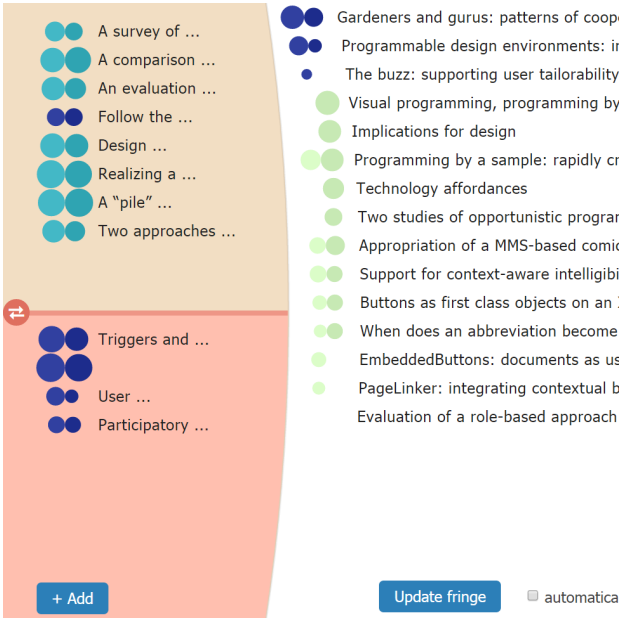

Fig. 9. A typical view of the interface after a user has been doing their literature review. Some papers are in the To Read list, and some have already been read.

10.2 Scenario 2

Fotis is in the process of doing a literature review for his CS544m Project at UBC. He heard great things about a website called PaperQuest so he is using it to help him find interesting papers. He has already read a couple and has selected two more to read (Figure 11), when suddenly he notices a recommendation for a paper very high up on the list that strangely has few citations. Curious to understand what is going on he displays the paper's links, and sees that "The buzz: supporting user tailorability in awareness applications" cites 4 of the papers he has already read and also the 2 that are in his To Read list (Figure 12a).

Fig. 10. Scenario 1: navigating through the interface. a) Top left: a few papers are moved from the To Read list to the Core. b) Top right: the Fringe is updated and some papers selected. c) Bottom left: papers in the Fringe are expanded with semantic zoom to read their abstracts. d) Bottom right: papers are moved to the To Read list.

To clarify which papers they are, he drags the Fringe border to the right so he can read the paper titles (Figure 12b). His curiosity piqued, he decides to add the suggested paper to his To Read list (Figure 12c), after which he expands it to see more details and start reading the abstract (Figure 12d).

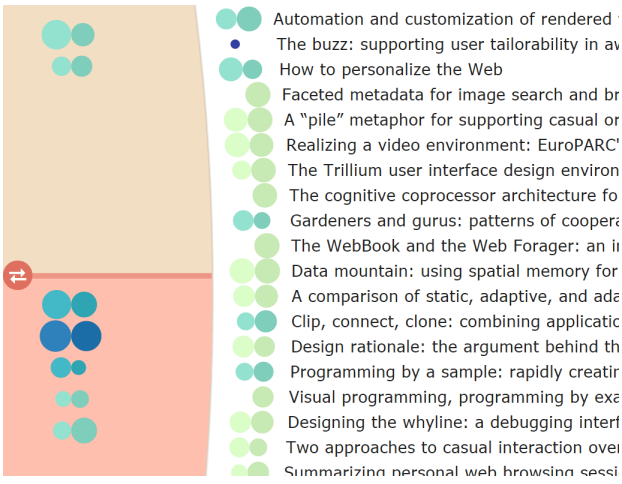

Fig. 11. A few papers have been added to the Core and to the To Read lists. The third paper in the Fringe is odd because it has high relevance but low citation count.

11 Discussion

Based on our own interactions and preliminary reactions by colleagues, we believe there is good potential to make this a useful tool. Our layout seems to support the common literature review workflow well, and presents a lot of information together, which at least seems to make the decision process more efficient. Ideally we would like to have not only efficiency, but also added value, in the sense that a researcher could identify relevant papers with our tool that they would

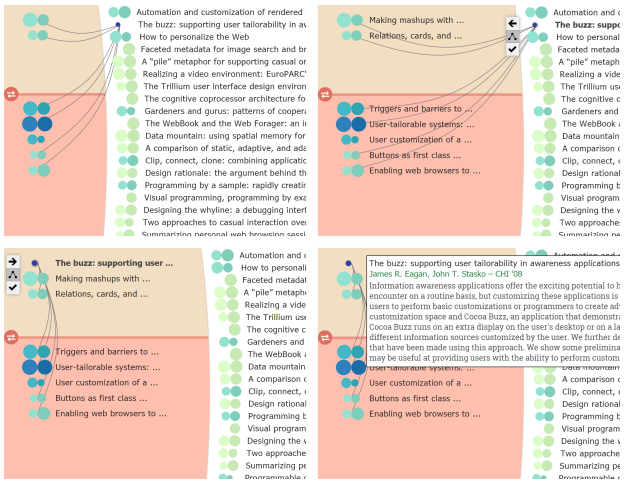

Fig. 12. Scenario 2: understanding the relevance of a paper. a) Top left: showing the links of a paper to see its references and citations. b) Top right: dragging the Fringe border to see the titles of papers in the Core. c) Bottom left: moving the paper to the To Read list. d) Bottom right: expanding the paper's details to read the abstract.

miss otherwise. We think the encoding choices we made promote this goal, but currently have no data to support this.

The biggest weakness we have identified is a problem with scale. Each of our views supports up to a few dozen papers, mostly because we are showing titles in a font size that should be comfortable to read, which takes up considerable space. While this can be partly improved by adding scrolling in the views, that idea does little to help showing links to papers in the hidden part of the list, or an overview of the data. Aggregation of elements should be considered, with the option to expand the aggregated elements to see them in more detail at the user's request.

We think the approach we took to this problem works, even if there are still things to fix and more functionality to implement. Having all relevant information together when making a decision of what to read next is extremely convenient, and enabling browsing of the full publication space in a domain facilitates discovery. We found out that many people conduct literature review in the same way, which was unexpected because most of them developed a process by themselves. Our interface supports this process naturally, and possibly extends it in useful ways.

The tools chosen for our implementation were appropriate and extremely helpful. D3 simplifies many processes that would otherwise be burdensome to implement, and as the system grows it also eliminates much complexity that would make the code fragile. We learned that the best way to see if an idea worked, when we were not sure about it, was to implement it and test it directly. This usually could be done quickly and promoted iteration, which we believe leads to better design.

12 Future Work

We have many ideas of improvements and extensions of our visualization. We would like to test more formally if the glyph encoding is valuable to users, and if together with the links encoding they have enough information to speed up their decision process of what to read. Having them together in a single visualization is one of our main contributions, so it would be interesting to know if they are helpful or not.

Improved filters and linked highlighting are the natural next steps in our design, as they would add functionality to the side views. Relevant papers should be highlighted when users hover over author names, or the publication years in the histogram. Selecting a few authors could filter the results of all views to show only their papers, and changing the limits of the year range could give the user only publications for that period. In this way, a user that identifies they have read few recent papers could focus only on those and fill a gap in their literature review. Even enabling regular scrolling in the Fringe view could be used as a simple filter: if selected papers are kept around, but others are scrolled out of view, the user could move to the bottom of the recommendations list to explore more results.

The relevance algorithm also needs more work. A machine learning approach to set the parameters could be very useful. A short-term alternative is to give control to advanced users so that they can tweak the parameters of the algorithm themselves. This could be achieved by clarifying the contents of the statistics page to users and making it more functional, so that they could determine their own cut-off thresholds and other normalization values. A slider in the main view could also be used to dynamically set the relative weight of citations counts and the connectedness measure in the computation of the relevance score.

Adding more sensemaking functionality would be an interesting challenge. This includes giving users the ability to give simple ratings to the papers they read, at a minimum starring a paper or discarding it. Additionally we would like to extend the functionality in the Core by allowing users to see a full network overview of the papers they have read and how they connect, with the possibility to label papers and to group them into meaningful categories.

To address the scale problem we would explore the idea of aggregating all links between the relevant paper and different *regions*, using line thickness to encode the number of links that are being aggregated. Besides aggregation, it would be interesting to design other representations of papers that use less pixels. While this is difficult to do in the fringe, where full paper titles cannot be removed without impacting the decision process of the user, we could use categories to cluster similar papers together, and show only one or two main keywords for a paper so that the user can remember which one it is. For example, after reading our report, it might be easily recognized by just the "PaperQuest" keyword instead of the full title. In this way we could at least offer more room for papers in the To Read and Core sections.

More datasets is another important goal. We plan on updating the CiteVis dataset with paper abstract information and adding it to the tool. Including multiple datasets would not only make the tool useful for a wider audience, but also allow us to start playing with inter-dataset information, like references to publications in other domains. It will probably be necessary to change the way datasets are fed into the system, and instead use a proper database server to be able to scale up to many users and collect their usage information. This in turn would enable us to include things like recommendations based on what other people read.

13 Conclusion

We designed and implemented a tool to support the process of a literature review, and in particular the task of deciding which paper to read next. Our main contribution is in presenting in a clear way just as much information as useful, from a large network of papers with all their metadata. The three regions of the main view afford a multi-level decision process, from selecting papers on the Fringe, to adding them to a To Read list, to organizing them in the Core.

We use a variety of visualization idioms to achieve our goals. Responsive interactions and transitions help create a flexible environment suitable for browsing and discovery. A carefully designed glyph encodes information on each paper in the dataset, and provides a visible rationale for the decisions of our custom relevance algorithm. Curved links indicate connections between papers to easily identify references and citations. A sidebar shows other facets of the data, such as the most frequent authors and the years of publication, and will be used for linked highlighting and filtering in the future.

Our system has a scale limitation that needs to be addressed: only a few dozen papers can be displayed, which limits its usefulness for a full literature review. However, the first impressions of potential users have been positive, as they recognized the benefits of a smarter and richer exploration of the literature than what is currently available through search engines and digital libraries. We plan to improve

PaperQuest to make it more useful and robust, and deploy it for real-world usage.

ACKNOWLEDGMENTS

The authors wish to thank Justin Matejka and Autodesk Research for giving them access to the Citeology dataset.

REFERENCES

[1] K. Borner and L. Viswanath. Major Information Visualization Authors, Papers and Topics in the ACM Library. In *IEEE Symposium on Information Visualization*, pages r1–r1. IEEE, 2004.

[2] M. Bostock, V. Ogievetsky, and J. Heer. D: Data-Driven Documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–9, Dec. 2011.

[3] D. H. Chau, A. Kittur, J. I. Hong, and C. Faloutsos. Apolo: Making Sense of Large Network Data by Combining Rich User Interaction and Machine Learning. In *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*, page 167, New York, New York, USA, May 2011. ACM Press.

[4] M. Dork, N. H. Riche, G. Ramos, and S. Dumais. PivotPaths: Strolling through Faceted Information Spaces. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2709–2718, Dec. 2012.

[5] N. Elmqvist and P. Tsigas. CiteWiz: a tool for the visualization of scientific citation networks. *Information Visualization*, 6(3):215–232, Oct. 2007.

[6] G. W. Furnas. A fisheye follow-up. In *Proceedings of the SIGCHI conference on Human Factors in computing systems - CHI '06*, page 999, New York, New York, USA, Apr. 2006. ACM Press.

[7] P. Isenberg, T. Isenberg, M. Sedlmair, J. Chen, and T. Möller. Toward a deeper understanding of Visualization through keyword analysis. Aug. 2014.

[8] H. Kang, C. Plaisant, B. Lee, and B. B. Bederson. NetLens: iterative exploration of content-actor network data. *Information Visualization*, 6(1):18–31, Jan. 2007.

[9] B. Lee, M. Czerwinski, G. Robertson, and B. B. Bederson. Understanding research trends in conferences using paperLens. In *CHI '05 extended abstracts on Human factors in computing systems - CHI '05*, page 1969, New York, New York, USA, Apr. 2005. ACM Press.

[10] J. D. Mackinlay, R. Rao, and S. K. Card. An organic user interface for searching citation links. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '95*, pages 67–73, New York, New York, USA, May 1995. ACM Press.

[11] J. Matejka, T. Grossman, and G. Fitzmaurice. Citeology: visualizing paper genealogy. In *Proceedings of the 2012 ACM annual conference extended abstracts on Human Factors in Computing Systems Extended Abstracts - CHI EA '12*, page 181, New York, New York, USA, May 2012. ACM Press.

[12] D. M. Russell, M. J. Stefik, P. Pirolli, and S. K. Card. The cost structure of sensemaking. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '93*, pages 269–276, New York, New York, USA, May 1993. ACM Press.

[13] J. Stasko, J. Choo, Y. Han, and M. Hu. Citevis: Exploring conference paper citation data visually. *Proceedings of the IEEE Conference VIS 2013*, pages 2–3, 2013.

[14] J. Stasko, C. Görg, and R. Spence. Jigsaw: supporting investigative analysis through interactive visualization. *Information Visualization*, 7(2):118–132, 2008.

[15] S. van den Elzen and J. J. van Wijk. Multivariate Network Exploration and Presentation: From Detail to Overview via Selections and Aggregations. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2310–2319, Dec. 2014.

[16] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers, second edition, 2004.

[17] K.-P. Yee, K. Swearingen, K. Li, and M. Hearst. Faceted metadata for image search and browsing. In *Proceedings of the conference on Human factors in computing systems - CHI '03*, page 401, New York, New York, USA, Apr. 2003. ACM Press.

[18] X. Zhang, Y. Qu, C. L. Giles, and P. Song. CiteSense: Supporting Sensemaking of Research Literature. In *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08*, page 677, New York, New York, USA, Apr. 2008. ACM Press.