How Hierarchical Topics Evolve in Large Text Corpora

Weiwei Cui, *Member, IEEE*, Shixia Liu, *Senior Member, IEEE*, Zhuofeng Wu, Hao Wei

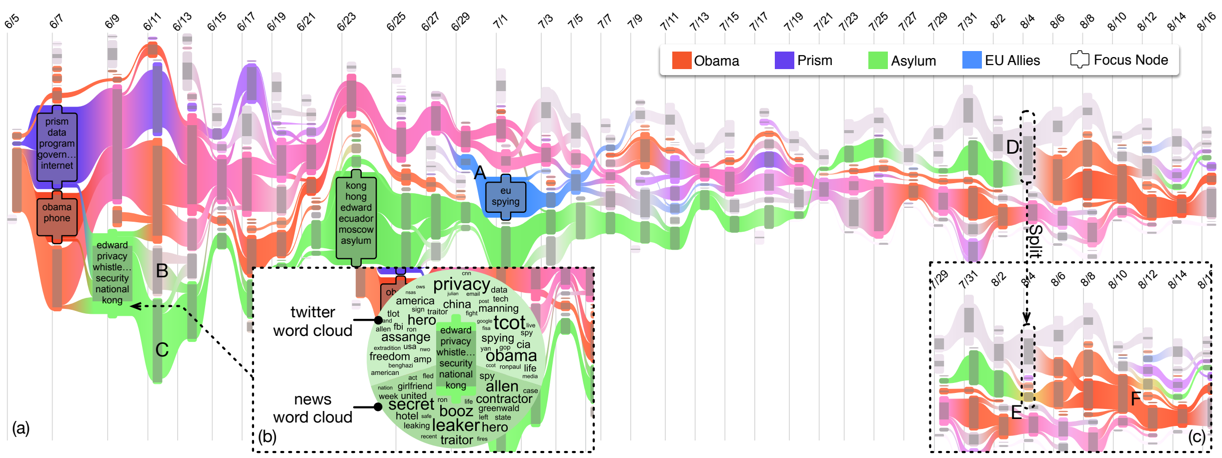

Figure 1. *RoseRiver*, a visual analytics system for exploring evolutionary hierarchical topics. (a) Overview of the Prism scandal (Jun. 5 to Aug. 16, 2013). Four colors represent the four major topics. Topics are displayed as vertical bars. The color stripes represent the evolving relationships between topics. (b) Comparison of the prominent keywords in tweets and news articles of the topic. The arc lengths encode the news article and tweet numbers (in log scale). (c) The new layout generated by splitting the gray topic.

Abstract— Using a sequence of topic trees to organize documents is a popular way to represent hierarchical and evolving topics in text corpora. However, following evolving topics in the context of topic trees remains difficult for users. To address this issue, we present an interactive visual text analysis approach to allow users to progressively explore and analyze the complex evolutionary patterns of hierarchical topics. The key idea behind our approach is to exploit a tree cut to approximate each tree and allow users to interactively modify the tree cuts based on their interests. In particular, we propose an incremental evolutionary tree cut algorithm with the goal of balancing 1) the fitness of each tree cut and the smoothness between adjacent tree cuts; 2) the historical and new information related to user interests. A time-based visualization is designed to illustrate the evolving topics over time. To preserve the mental map, we develop a stable layout algorithm. As a result, our approach can quickly guide users to progressively gain profound insights into evolving hierarchical topics. We evaluate the effectiveness of the proposed method on Amazon's Mechanical Turk and real-world news data. The results show that users are able to successfully analyze evolving topics in text data.

Index Terms—Hierarchical topic visualization, evolutionary tree clustering, data transformation.

1 Introduction

To effectively handle huge numbers of documents on the Web, new efforts have been made to organize text corpora using evolving multi-branch trees [37, 43], known as topic trees. This approach has two merits. First, it provides very interpretable topic results, since real-world text corpora are naturally organized by multi-branch topic trees [5, 23, 37] and the topic trees often evolve over time [37]. Second, hierarchical structures provide an intuitive way to navigate topics, from a global overview to local details. As a result, this approach addresses aspects of the scalability issue and introduces increased flexibility to text data management. However, even with coherent topic trees over time, the exploration and consumption of evolving topics in the context of hierarchies remain difficult.

There are two challenges to understanding evolving topic trees. First is the challenge of effectively presenting various topic evolution patterns to support analysts with their tasks. In a topic tree sequence generated by an evolutionary tree clustering model [37], a topic in the current tree may be related to a number of topics in the previous tree, which may not be at the same tree level. By linking the related topics across trees (topic mapping), we provide an overview of topic evolution in the context of tree structures. However, the direct presentation of topic mapping and parent-child relationships for each tree is overwhelming and causes cognitive overload for users. For example, it is difficult for users to follow topic alignments across trees because of the number of links presented. Therefore, it is desirable to allow users to obtain a full picture of the data and then quickly focus on the information of interest. Second is the challenge of preserving a meaningful context for progressive navigation. A better understanding of hierarchically evolving topics is not a one-click task. The user often switches back and forth among different topics and drills in to examine their children. If the context changes significantly, the user can easily get lost. As a result, a meaningful context that balances historical and new information related to user interests is preferred.

The state-of-the-art approach, *TextFlow* [10], assumes the evolving topics are flat. Thus, its visualization cannot be directly employed to illustrate evolving topic trees. To solve this problem, we can extract topic alignments at the same levels across trees and regard the topic alignments at different levels as different topic sets. Afterwards, *TextFlow* can visually convey the topic sets one-by-one to users. However, it is hard for users to form a comprehensive picture of hierarchical topic evolution since many topic alignments are not at the same level.

To address these issues, we present an interactive visual text anal-

- *W. Cui and S. Liu are with Microsoft Research. S. Liu is the correspondence author. E-mail: {weiwei.cui, shixia.liu}@microsoft.com.*
- *Z. Wu is with Nankai University. Email: leewingsac@gmail.com.*
- *H. Wei is with Zhejiang University. Email: sherryweihao@gmail.com.*

Manuscript received 31 Mar. 2014; accepted 1 Aug. 2014; date of publication xx xxx 2014; date of current version xx xxx 2014. For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

ysis approach, *RoseRiver*, which allows users to progressively explore and analyze the complex evolution patterns of hierarchical topics. First, we propose an incremental evolutionary tree cut algorithm that considers three factors. The first factor exploits a tree cut to approximate each tree and allows users to interactively modify tree cuts to refine the topics displayed for exploration. The second factor balances the fitness of each tree cut and the smoothness between adjacent tree cuts. Thus, users can focus on their interests with a coherent view. The last factor balances historical and new information related to user interests, so that users can progressively gain profound insight into evolving hierarchical topics without losing context. We combine the three factors in an optimization model, which is inspired by the graph cut technique [17] in computer vision. On top of the incremental evolutionary tree cut algorithm, a time-based visualization is designed to illustrate the evolving topics over time. To ensure a smooth exploration experience, a stable layout algorithm is also developed to preserve the mental map. The main contributions of this work are:

- An incremental evolutionary tree cut algorithm that satisfactorily balances the fitness of each tree cut and the smoothness between adjacent tree cuts, as well as historical and new information related to user interests, in an optimization model.
- A visual analysis system that connects large text corpora with people by effectively presenting interesting topics and their evolution over time to users in an intuitive and manageable manner.

2 Related Work

Visually analyzing evolving topics in text corpora has become a widely researched topic over the past few years [10, 14, 18]. Numerous methods [14, 42, 41] leverage a river metaphor to convey evolving topics over time. For example, ThemeRiver [14] visually depicts the changes in keyword strength in a text corpus over time using a river metaphor. Xu et al. [42] and Wu et al. [41] combined the river metaphor with other visualizations to study the spreading patterns of topic and sentiment in social media. TIARA [21, 22] tightly integrates interactive visualization with topic analysis techniques to assist users in understanding a document collection. In particular, it employs the LDA model [4] to analyze a large text corpus and visually illustrates the evolution of topics using an enhanced stacked graph. MemeTracker [16] was developed to effectively identify a huge number of topics from millions of news articles. This provides a coherent representation of the news cycle, allowing users to track the temporal behaviors of memes represented by short phrases in news and blogs. TextFlow [10], which models the relationships between evolving topics, assists analysts in the visual analysis of topic merging/splitting relationships and tracks their evolution over time.

Recently, several visualization techniques were proposed to help users analyze temporal events and their evolving patterns. EventRiver [24] assumes that clusters of news articles with similar content are adjacent in time and can be mapped to events. With this assumption, this method automatically detects and presents interesting events to reveal their impact over time. LifeFlow [39] and Outflow [38] facilitate the exploration of temporal event sequences. To achieve this, they aggregate multiple event sequences into a tree and a directed acyclic graph, respectively. Afterwards, a timeline visualization is used to display the aggregated event sequences from multiple aspects. Monroe et al. [26] developed a set of user-driven data implication techniques to help users understand large temporal event records.

The above approaches focus on the visual exploration of evolving topics/events with flat structures. To the best of our knowledge, our work is among the first to support the visual analysis of evolving hierarchical topics over time.

Inspired by XKCD's movie narrative charts [27], a storyline visualization has been developed to illustrate the temporal interactions between characters in a story. Ogawa and Ma [29] developed a set of heuristic rules to quickly generate a storyline layout. Although this method can generate a storyline layout in real-time, it may result in a layout with many line crossings and wiggles. To overcome this limitation, Tanahashi and Ma [35] formulate the storyline layout as an optimization problem. They leverage spatial information to convey hierarchical relationships among characters. However, hierarchical information is not considered in the layout algorithm and is added in the post-processing step instead. As a result, it cannot leverage the location hierarchy to handle a large number of character lines. Liu et al. [20] proposed an efficient optimization approach to support real-time interaction. They formulate the storyline layout as a novel hybrid optimization approach that combined discrete and continuous optimization. This approach utilizes a predefined location hierarchy to handle thousands of entities in the storyline layout. HierarcicalTopics [11] and TopicPanorama [19] employ the BRT model [5] to construct a static topic hierarchy from a set of topics, which is used to present changes in the topic content in a hierarchal manner.

Although the aforementioned methods can manage scalability by utilizing a static hierarchy, they may fail to concisely reflect the new topic structure as new documents appear [37]. Our method differs from these approaches as it generates a set of evolving topic trees, which are used to smoothly organize a large number of topics over time. To better present evolving hierarchical topics on limited screen real estate, we propose an incremental evolving tree cut algorithm to extract an appropriate number of topics for each tree based on the user selected focus node(s).

3 RoseRiver

3.1 Design Consideration

This research problem was identified in our collaborations with domain experts. In the past two years, over ten experts approached us for advice on analyzing large text corpora. Several of their requirements, regardless of the various scenarios, shared a common objective, namely, to understand the major content of a large text corpus and its evolution patterns over time. Thus, *TextFlow* [10], which was designed for analyzing topics and their merging/splitting relationships over time, was introduced to the experts. Once their data was loaded into *TextFlow*, the experts were impressed by the informative and intuitive visualization and immediately discovered interesting patterns, which either echoed their previous discoveries or triggered new ideas.

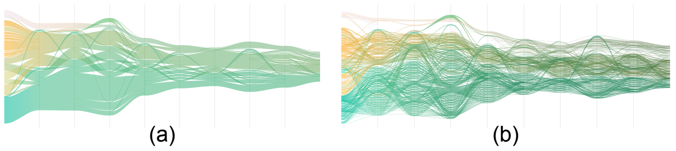

Figure 2. *TextFlow* representations of the first ten time points of the Prism dataset: (a) using the first level topics in each topic tree; (b) using leaf topics in each topic tree. The corresponding *RoseRiver* representation is displayed in Fig. 14.

However, according to the experts, the system still possesses two limitations that hamper its adoption and wide use. First, *TextFlow* can only generate a flat topic structure, which can cause a dilemma when handling large document collections. If the topics are modeled at a coarse level and dozens of topics are generated, *TextFlow* can manage them very well. However, the topics would be too general and there is not enough detail presented for users to fully understand the text corpora (Fig 2(a)). In contrast, if the topics are modeled at a finer level, too many topics will be produced, and *TextFlow* may fail to provide a clear overview (Fig 2(b)). Second, although *TextFlow* can generate several topic sets from a coarse level to a fine level and visually convey them one-by-one to the user, mentally connecting the topic sets can be hard for users because these are learned separately.

To this end, we worked closely with four experts, including one public relations manager, one professor in environmental politics, and two professors in media and communications. Following the *nested model for visualization design and validation* [28], we conducted a series of interviews and focus-group sessions with the four experts to generalize and develop an understanding on how they explore and analyze hierarchical topics in a large document collection. Based on their feedback and previous graph visualization related research, we derived the following design guidelines.

Why evolving topic trees? An overview is essential for analysis tasks in many expert applications. Previous research showed that users can better understand and consume information if they are provided with hierarchical topic information [43]. As a result, we decided to build a sequence of evolutionary topic trees to organize documents at different times, which can naturally solve the two problems above. First, a topic tree can adequately describe the document distribution at each time, as well as provide different overview levels based on user interests. Second, each tree in the sequence is similar to the one at the previous time point, and the related topics between adjacent topic trees are mapped together.

Why evolutionary tree cuts? In an early version of *RoseRiver*, we presented topic trees in small multiple views and used lines to represent topic mappings between adjacent trees. However, the experts experienced difficulty in their explorations because of the *"visual clutter and unnecessarily overwhelming details."* Thus, a preferable option is to extract a set of representative topics to describe each tree. Although we can select the representative topics tree-by-tree based on user interests [7], this may fail to provide a coherent view over time. To solve this problem and preserve the mental map of users, we developed an evolving tree cut algorithm.

Why incremental evolutionary tree cuts? A better understanding of evolving topics in the context of topic trees is not a one-click deal. It is an iterative and progressive exploration and analysis process. According to our target users, they often examine different topic nodes and drill in each of them for detailed comparisons in their analysis tasks. For each exploration, a new set of tree cuts are generated. The experts claim that their analysis tasks can benefit from a meaningful context that balances historical and new information related to their current interests. The familiar context provided by the historical information will guide them as they search for new patterns. As a result, an incremental evolutionary tree cut algorithm to generate a set of stable tree cuts is desirable.

Why a stable layout? A generally desirable criterion of dynamic data visualization is preserving the mental map, namely, keeping stable layouts across time points [2, 3]. We followed this design guideline in our visualization. When interacting with the *RoseRiver* visualization, the placement of existing nodes and edges was minimally modified compared with the previous layout. With this feature, users can easily keep track of nodes and edges in various interactions and easily find interesting patterns, which was confirmed by our domain experts.

Why a familiar visual metaphor? A familiar visual metaphor should be used to help users quickly understand unfamiliar information as it can lower the cognitive load imposed on a user and increase the rate of comprehension [25, 34]. When we introduced *TextFlow* to domain experts, they affirmed that the river metaphor is intuitive and easy to remember. Thus, we adopted the river metaphor again in *RoseRiver* to reduce the cognitive load.

3.2 System Pipeline

Fig. 3 briefly illustrates our system pipeline. First, in the data module, a sequence of topic trees is extracted from the input documents by our evolutionary tree clustering technique. The technique aims to improve the organization of the topics at different time points, as well as map related topics between adjacent topic trees. Second, the tree-cut module transforms the tree clustering results into a set of salient topics based on user interests. Third, in the visualization module, the transformed results, including the topic nodes of each tree cut, matched topics and documents, are visualized for further exploration. Finally, feedback on the tree-cut module from users incrementally refines the salient topics through interaction.

4 Evolutionary Topic Tree

In *RoseRiver*, we employ evolutionary multi-branch tree clustering [37] to generate a sequence of coherent topic trees, which ensures the fitness of each tree and the smoothness between adjacent trees are well balanced Specifically, given a document collection, the evolutionary tree clustering method generates the following outputs.

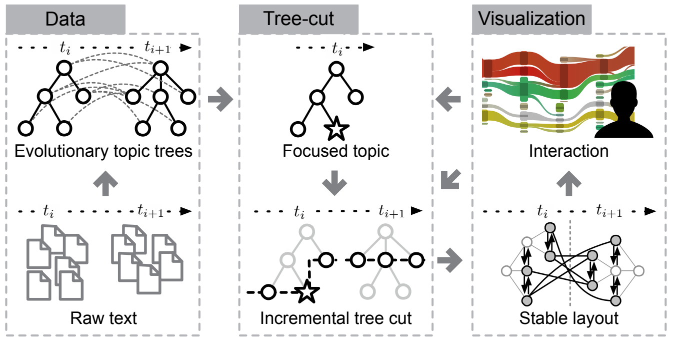

Figure 3. *RoseRiver* system consists of three components: (a) data, (b) tree-cut, and (c) visualization modules.

- A sequence of evolving **topic trees**, in which the interior node of a topic tree is a topic and the leaf node is a document. Documents that directly belong to an interior node also form a topic whose parent is the interior node during post-processing, such that each document is a leaf node. In the topic trees, each document only belongs to one leaf topic and its ancestor topics.
- A set of **topic pairs** between adjacent topic trees. For two adjacent topic trees, the clustering model maps the related topics, such as similar topics and topics with a splitting/merging relationship, together. We call these two mapped topics a topic pair.
- A group of **document pairs** within each topic pair. The model also maps two similar documents together, each of which is from each of the topic pairs. These documents are referred to as a document pair.

5 Incremental Evolutionary Tree Cut

The evolutionary tree model builds a sequence of topic trees, which often consists of a large number of topics and trees. As a result, directly visualizing all the trees and their evolution patterns easily overwhelms users and cause them to lose focus. To solve this issue, our system transforms the evolutionary tree clustering results and extracts the most salient content to be visualized at each given time point.

5.1 Basic Idea

To extract a set of nodes that can represent a topic tree based on user interests, we introduce a tree cut concept into *RoseRiver*. A **tree cut** is defined as a set of nodes such that every path from the root of the tree to a leaf will contain exactly one node from the cut. Thus, each cut corresponds to a set of representative nodes. Fig. 4(a) shows an example of a tree cut. In the following discussion, we refer to each node on the tree cut as a "**cut node**."

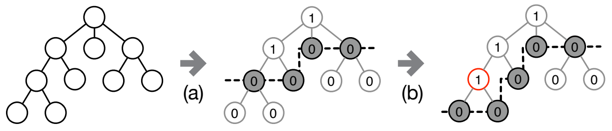

Figure 4. Tree cut generation and modification: (a) The cut nodes are dark grey. The nodes above the cut are labeled 1, whereas the rest are 0. (b)The label of the red topic is changed to 1. Thus the tree cut is modified accordingly.

The goal of our algorithm is to determine a desirable cut for each tree based on user interests and to allow users to progressively explore and refine topic sets without losing context. In our system, user interests are obtained by specifying one or more topic nodes (focus nodes) in the topic trees. A straightforward method is to apply the degree-of-interest (DOI) technique [7, 36] on every tree separately. Although this method usually obtains a better topic summarization (better fitness) for each time, the smoothness of tree cuts across multiple trees can be lost. Thus, tracking topic changes over time is difficult for users.

In contrast, overlapping successive views [40] is a frequently adopted principle when processing temporal data [8]. In *RoseRiver*, we consider both topic mappings between trees and tree cut results in the previous exploration steps to ensure a smooth exploration experience. Inspired by the graph cut techniques [17], we formulate the tree cut generation and refinement process as an optimization-based labeling algorithm, in which the topic nodes above the tree cuts are labeled 1 and the rest (including the cut nodes) are 0 (Fig. 4).

The algorithm involves four steps: 1) users specify one or more topic nodes as the focus node(s); 2) the key tree cut(s) is generated based on the user selected focus topic(s); 3) a set of evolving tree cuts are derived based on the key tree cut(s); 4) users interact to refine the tree cut results without losing context. We take one focus node as an example to illustrate the evolving tree cut algorithm.

5.2 Key Tree Cut

Given a focus node and the topic tree (key topic tree) it belongs to, we employ the DOI to generate a key tree cut. In Furnas's work [12], the DOI of a data item x with regard to a given focus y, is defined as:

$$\mathbf{DOI}(x|y) = \mathbf{API}(x) - \mathbf{D}(x,y), \tag{1}$$

where **API** is a priori interest function that defines the general importance of x irrespective of y. **D** is the distance between x and y, which measures the interest of x depending on focus y.

Furnas also showed that there are natural **DOI** definitions for many types of data [12]. For example, in a tree, the node depths can be used for **API** and the path length distance between x and y in the tree can be used for **D**. However, these two terms still need tuning based on different applications. Starting with the natural definitions proposed by Furnas, we tested several alternatives for **API** and **D**, such as the number of documents in a topic node, the degree of the topic node, and content similarity between topics. Based on the evaluation results, we have adopted two factors, the path length distance and content similarity. Accordingly, we define the **DOI** as:

$$\mathbf{DOI}(T_x|T_{focus}) = \mathbf{API}(T_x) - (\beta_1 \mathbf{D_T}(T_x, T_{focus}) + \beta_2 \mathbf{D_C}(T_x, T_{focus})), \tag{2}$$

where $\mathbf{API}(T_x) = -1/\mathbf{D_T}(T_x, root)$, T_{focus} is the given focus topic node, $\mathbf{D_T}(T_x, T_{focus})$ is the tree distance between T_x and T_{focus}, and $\mathbf{D_C}(T_x, T_{focus}) = -\mathbf{S}(T_x, T_{focus})$ is the content distance between T_x and T_{focus}, which takes the negative value of the cosine similarity between T_x and T_{focus}. In our implementation, $\beta_1 = 1$, $\beta_2 = 3$.

5.3 Incremental Tree Cut Propagation

Once the key tree cut is derived, we propagate it to other topic trees to generate a set of coherent tree cuts. The nodes in the key topic tree already contain labels. Thus, the major objective of the tree cut propagation algorithm is to assign labels to the nodes in other topic trees and make the resulting tree cuts smooth and coherent. To achieve this goal, three factors need to be balanced:

- The topic content to ensure the tree cuts reflect user interests.
- The topic relationships produced by the evolutionary tree model to ensure that the resulting tree cuts are smooth over time.
- The previous node labels (if any) to ensure that the tree cut results do not change dramatically during the exploration.

To this end, we use four metrics: content similarity, tree structure, topic mapping between adjacent trees, and label changes between two sets of tree cuts. The first three metrics are inspired by the graph cut technique [17] and the fourth one is designed for *RoseRiver* to preserve stability between adjacent sets of tree cuts. The four metrics are combined into the following energy function:

$$\mathbf{E}(X) = \sum_{T_r \in \mathscr{T}} \mathbf{E}_1(x_r) + \lambda_1 \sum_{(T_r,T_s)\in \mathscr{A}_t} \mathbf{E}_2(x_r,x_s) + \lambda_2 \sum_{(T_r,T_s)\in \mathscr{A}_m} \mathbf{E}_3(x_r,x_s) + \lambda_3 \sum_{T_r \in \mathscr{T}} \mathbf{E}_4(x_r), \tag{3}$$

where $X = \{x_r : \forall r\}$, x_r is the label (1 or 0) of topic T_r, $\mathscr{T}$ represents all the topics in the tree sequence, $\mathscr{A}_t$ denotes all the node pairs with parent-child relationships, and $\mathscr{A}_m$ represents all the topic pairs between adjacent trees, which are mapped together by the evolutionary tree clustering model. Parameters λ_1, λ_2, and λ_3 balance the four metrics. In our implementation, $\lambda_1 = 1, \lambda_2 = 0.5$, and $\lambda_3 = 1$.

Similarity Energy $\mathbf{E}_1$ is a content metric that measures the content similarity of topic T_r to the two topic sets, $\{1\}$ and $\{0\}$. $\{1\}$ and $\{0\}$ represent the nodes with fixed labels as 1 and 0. For a topic T_r, cosine similarity $\mathbf{S}(T_r, T_s)$ is used to compute the similarity value between T_r and each topic T_s in $\{1\}$ or $\{0\}$. Particularly, $\mathbf{E}_1(x_r)$ is defined as:

$$\mathbf{E}_1(x_r) = \begin{cases} 0, & \text{if } T_r \in \{x_r\}; \\ \infty, & \text{if } T_r \in \{1 - x_r\}; \\ \min_{T_r \in \{x_r\}}(-\log(\mathbf{S}(T_r,T_s))), & \text{otherwise.} \end{cases}$$

Tree Structure Energy $\mathbf{E}_2$ is a structure metric based on the parent-child relationship. This penalizes the label differences between the parent-child node pairs, and thus encourages these pairs to possess the same label. We only penalize the case in which the parent label x_p is 0 and the child label x_c is 1.

$$\mathbf{E}_2(x_p,x_c) = (1-x_p)|x_c - x_p|. \tag{4}$$

Topic Mapping Energy $\mathbf{E}_3$ is a metric based on the topic mapping between adjacent topic trees. This metric penalizes the label differences between topic pairs that are mapped by the clustering model and encourages the topic pairs to possess the same label.

$$\mathbf{E}_3(x_r,x_s) = |x_r - x_s|\omega(x_r,x_s), \tag{5}$$

where $\omega(x_r,x_s)$ denotes the mapping weight computed by the evolutionary tree clustering model.

Label Change Energy $\mathbf{E}_4$. Although a sequence of tree cuts are initialized through metrics $\mathbf{E}_1$, $\mathbf{E}_2$, and $\mathbf{E}_3$, users often need to refine them during the exploration. For example, a user may find a cut node interesting and decides to explore more detailed topic nodes. Thus, he or she manually labels the cut node as 1 and lower the corresponding tree cut at that cut node (Fig. 4(b)). If we only change the tree cut that the user modified and leave the others unchanged, the user may fail to get the right context for further analysis. One naive solution is to allow the user to manually modify other tree cuts. However, this requires knowledge and is very time-consuming, especially when there are hundreds of topic trees, each of which contains hundreds of or even thousands of nodes. Another solution is to add the cut node to $\{1\}$ and re-run the optimization model with $\mathbf{E}_1$, $\mathbf{E}_2$, and $\mathbf{E}_3$ to generate a new sequence of tree cuts for users to explore. However, this method does not maintain the stability between the old and new sets of tree cuts.

To solve this issue, we introduce one more metric, $\mathbf{E}_4$, into the optimization model to maintain the stability between the old and new tree cut results in the interactive exploration. This metric encourages topic T_r to acquire the same label in the old and new tree cut results.

$$\mathbf{E}_4(x_r) = |x_r - x'_r|, \tag{6}$$

where x'_r is the label of T_r in the previous tree cut result.

The energy function **E** is globally minimized by the graph cut algorithm [6]. The optimization may fail to derive the optimal tree cuts when a parent node is labeled 0 and at least one of its child nodes is labeled 1. To solve this problem, we employ the following strategy in the optimization: if a node is labeled 1, all its ancestors are labeled 1.

6 VISUALIZATION

In *RoseRiver*, we focus on studying evolution patterns such as topic birth, death, splitting, or merging at different levels of granularities based on the tree structure. Fig. 1(a) shows an example of the visualization, where the x-axis represents time. The nodes on each tree cut are displayed as vertical bars, which are arranged vertically at the corresponding time point. The color stripes represent the evolution relationship between cut nodes.

6.1 Visual Encoding

Tree Cut. Each cut node is represented by a vertical bar. Tree depth is encoded by the horizontal offset to the time point. A node deeper in the tree will be moved further to the right. To improve space efficiency, the mapping between the depth and offset is non-linear: $\mathbf{B}(r) = W\sum_{i=1}^{d_r}(\frac{1}{2})^{i-1}$, where d_r is the depth of topic T_r in the corresponding tree, and W is the width of the topic bar. Fig. 5 shows an example of the depth encoding scheme. In our application examples, we found that the topic trees are not very deep, with a typical depth from 4 to 6. Thus, we directly use $\mathbf{B}(r) = W\sum_{i=1}^{d_r}(\frac{1}{2})$.

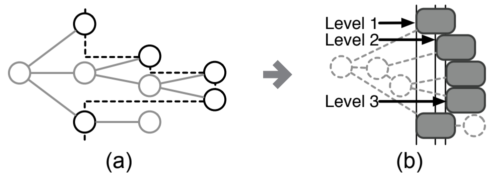

Figure 5. Visual encoding of depth information: (a) the tree cut result; (b) cut nodes are aligned based on their depth information.

Document. The height of a topic bar indicates the number of documents it contains. The color stripe between two topic bars indicates the number of document pairs between them. For example, the left width of the stripe represents the documents that are mapped to the documents in the right topic bar. The dark region (Fig. 6) inside each topic bar represents the documents that are mapped to the documents both in its previous and next topic trees. The height of the dark region encodes the portion of such documents.

Users can get an overview of the hierarchical topic evolution patterns by examining the shape changes of the color stripes and the dark regions in the bars, and then explore the content further to examine what triggers and contributes to such patterns. Fig. 6 illustrates several evolution patterns.

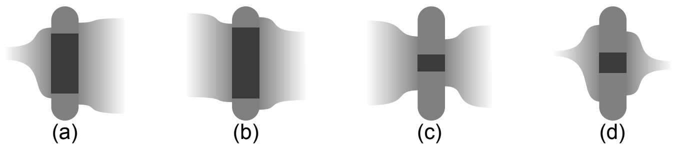

Figure 6. Four example patterns and their possible explanations: (a) a new topic is emerging; (b) a topic is still active but changes slowly; (c) a topic is active, but changes immensely; (d) a momentary topic emerges and disappears rapidly.

Degree of Interest of Topics. The focus topic is given a fully saturated unique color and a thick black border. The colors of other topics represent their similarity to the focus topic. The color gradually changes to gray with the decrease of the similarity value.

6.2 Stable Layout

The *RoseRiver* layout can be formulated as the layout of a directed acyclic graph (DAG). We introduce some definitions that are useful in subsequent discussions. Given a DAG graph $G = (V, E)$, V can be partitioned into disjointed sets $V = V_1 \cup V_2 \cup \cdots \cup V_m$ with $|V_i| = n_i$. In *RoseRiver*, V_i represents the cut node set at time point i. Similarly, $E = E_1 \cup E_2 \cup \cdots \cup E_{m-1}$, where E_i is the edge set between V_i and V_{i+1}. In our case, E_i is the topic mapping between time points i and $i+1$.

To improve the visual quality, various methods have been proposed to minimize the edge crossings in the DAG layout [33] (baseline). In *RoseRiver*, we employ an optimal method proposed in [15] to reduce the edge crossings . However, in our specific application, minimizing edge crossings is not the only goal of the layout algorithm. We also need to maintain the mental map during the exploration, which means that the layout should be incrementally updated according to every new set of cut nodes generated. Technically, the new layout must be as stable as possible with regards to the previous one. Based on the two above considerations, the layout model can be formulated as follows:

$$\min\Big(\sum_{t=1}^{m-1}\sum_{(a,b),(c,d)\in E_t}(u^t_{ac}u^{t+1}_{db}+u^t_{ca}u^{t+1}_{bd})+\alpha\sum_{t=1}^{m}\sum_{1\le i<j\le n_t} z^t_{ij}\Big), \tag{7}$$

subject to

$$u^t_{ij}+u^t_{ji}=1, \qquad \forall i,j\in V_t \text{ and } 1\le t\le m; \tag{7a}$$

$$0\le u^t_{ij}+u^t_{jk}+u^t_{ki}\le 2, \qquad \forall i,j,k\in V_t \text{ and } 1\le t\le m; \tag{7b}$$

$$u^t_{ij}\in\{0,1\}, \qquad \forall i,j\in V_t \text{ and } 1\le t\le m. \tag{7c}$$

Here u^t_{ij} is the sequential order of nodes i and j. z^t_{ij} indicates if i and j reverse their orders in the new layout. Assuming π'_t and π_t are the previous and current permutations of V_t, u^t_{ij} and z^t_{ij} are defined as:

$$u^t_{ij}=\begin{cases}1, & \text{if } \pi_t(i)<\pi_t(j);\\ 0, & \text{otherwise.}\end{cases} \qquad z^t_{ij}=\begin{cases}u^t_{ij}, & \text{if } \pi'_t(i)>\pi'_t(j);\\ u^t_{ji}, & \text{otherwise.}\end{cases}$$

In Eq. 7, the first term represents the total number of edge crossings. The second term is the number of node pairs with an inconsistent order between the old and new layouts. α is the parameter that balances the two terms. In our implementation $\alpha = 10$. The detailed constraints are illustrated below:

(7a) Order constraint that prevents an forbidden state in which u_{ij} and u_{ji} possess the same values.

(7b) Cycle constraint that prevents i, j, and k from forming a loop.

(7c) Domain constraint that defines the possible values of indicator u_{ij}. If i is placed before j, $u_{ij} = 1$, otherwise, $u_{ij} = 0$.

This nonlinear integer optimization problem can be easily converted into a linear integer problem using variable substitutions. The Mosek package [1] is utilized to determine the optimal solution. Fig. 10 demonstrates one result generated by our stable layout algorithm.

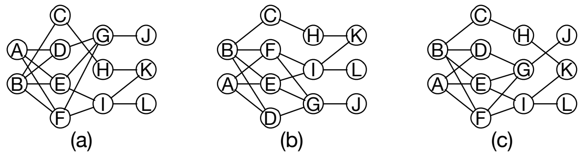

Figure 7. Example of stable layout: (a) the initial layout, (b) the optimized layout with minimum edge crossings, (c) the stable layout. The optimized layout has four edge crossings and eight pairs of nodes with the vertical order reversed. The stable layout has five edge crossing and two pairs of nodes with the vertical order reversed.

6.3 Interaction

RoseRiver allows users to interact with the clustering results to understand complex relationships and perform an in-depth analysis, as well as examine relevant data from multiple perspectives. In particular, we support several types of user interactions.

6.3.1 Details on Demand

Once users understand the overall hierarchical topic patterns, they will want more details on the topics to identify the main cause leading to the patterns of interest. To support these requests, *RoseRiver* extracts a set of distinctive keywords and representative documents for each topic.

Given a topic T, we rank all the documents $\mathbf{x}$ it contains using the cosine similarity measure $\mathbf{S}(\mathbf{x}, T)$, and choose the top-ranked documents. Related document titles are placed on a timeline to help users understand content evolution over time.

A set of distinctive keywords for each topic is selected to allow users to examine the keyword-based content, where distinctive keywords are those that occur frequently in the topic and seldom occur in others. Distinctive keywords are extracted based on the following criterion, which is adapted from the TF-IDF (term frequency-inverse document frequency) weighting scheme [32].

$$\mathbf{W}(w)^t_k = \mathbf{TF}(w)^t_k / \sum_k \mathbf{TF}(w)^t_k, \tag{8}$$

where w represents a word, $\mathbf{TF}(w)^t_k$ is the term frequency of w in topic T_k at time t.

The topic bar expands to show related keywords when a user hovers over it. To maintain the tree depth information, only the middle section of the bar is expanded and the two ends of the bar remain unchanged.

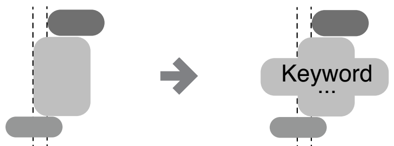

Figure 8. Example of node expansion: when a node is hovered over, it expands in the middle to show related keywords while maintaining the hierarchical information intact.

6.3.2 Collapsing/Expanding Time Points

Topic mapping relationships between different time points are not equally complex. Time intervals with complex topic mappings may contain important information. To help users identify critical time intervals, we adopt an entropy metric to measure complexity:

$$\mathbf{C}^{t,t+1} = -\sum_{i,j} \frac{|\mathcal{M}_{i,j}^{t,t+1}|}{|\mathcal{M}^{t,t+1}|} \log \frac{|\mathcal{M}_{i,j}^{t,t+1}|^2}{\sum_i |\mathcal{M}_{i,j}^{t,t+1}| \sum_j |\mathcal{M}_{i,j}^{t,t+1}|}, \qquad (9)$$

where $\mathcal{M}_{i,j}^{t,t+1}$ is the document pair set contained in (T_i^t, T_j^{t+1}), and $\mathcal{M}^{t,t+1} = \cup_{i,j} \mathcal{M}_{i,j}^{t,t+1}$. The larger the entropy value, the more complex the topic mapping at that time interval becomes. Based on the entropy values, our system collapses the time intervals that are less complex to scale with time. Alternatively, users can also collapse and expand time intervals based on their interests.

6.3.3 Splitting/Merging Topic Bars

Our data transformation may not always generate the results exactly as the users want. Thus, our system allows users to interactively change transformation results, i.e., splitting/merging topic bars.

When a user selects a topic bar that contains more than one topic node, our system expands that bar and divides it into several sub-bars representing its child topics. The height of each sub-bar is proportional to the document number in it. The color represents the DOI values relative to the current focus node. Afterwards, the user can select one of the sub-bars and move it out of the parent bar. The selected sub-bar becomes an independent topic bar in the visualization. Fig. 9 illustrates the splitting interaction.

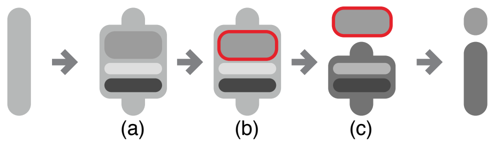

Figure 9. Example of splitting a topic bar: (a) expand the bar to see its child topics; (b) select a child topic to split; (c) visualize the selected child topic as a separate bar.

In addition, users can select two or more topic bars that represent sibling nodes and merge them into one topic bar, which will also be re-scaled and re-colored based on its new DOI value.

6.3.4 Changing Focus

Allowing users to quickly change their focus is an important feature of *RoseRiver*. Our system has methods to support different usage scenarios.

Using Search Box. This method is used when users know exactly what they want. With this method, users input the keywords that describe their interests. Our back-end Lucene engine [13] ranks the topic nodes based on their similarity to these keywords. After selecting one or more focus topics from the ranking list, our system runs the data transformation and generates the visualization accordingly.

Clicking. This method is used when users find an interesting topic through their exploration. During the exploration for deep analysis, the user may find an unexpected topic interesting and want to see more related topics. They can directly click that node and *RoseRiver* runs the data transformation and generates the visualization accordingly.

7 Evaluation

In this section, we evaluate our system based on three aspects. First, several quantitative evaluations are conducted on three datasets to show the effectiveness of the incremental evolutionary tree cut algorithm. After that, the advantage of the stable layout algorithm is demonstrated through three user studies on Amazon Mechanical Turk (AMT). Finally, we show the usefulness of the *RoseRiver* system through one real-world dataset and present the feedback from our domain experts.

7.1 Quantitative Evaluation

We implemented a baseline system based on the key tree cut generation method. A tree cut was specifically derived for each time via Eq. (2) in the baseline method. We performed experiments on three datasets to compare the smoothness and fitness of the results generated by our evolutionary tree cut algorithm and the baseline method.

- **Publication dataset *A*** that contains 3,860 papers on data mining and visualization (2000 to 2010). Papers were organized into 11 topic trees by year. The tree depths varied from 3 to 4, the total node numbers changed from 32 to 82, and the node number of the first level ranged from 10 to 33.
- **Prism dataset *B*** that includes 69,867 news articles and 568,225 tweets which were related to the "NSA prism spying scandal" (Jun. 3, 2013 to Feb. 9, 2014). Articles and tweets were put together and organized into 36 topic trees by week. The tree depths varied from 3 to 9, the total node numbers changed from 19 to 165, and the node number of the first level ranged from 3 to 20.
- **News dataset *C*** that includes 1,815,712 news articles from 51 news sources such as the New York Times, Reuters, and the BBC (Dec. 2, 2013 to Dec. 30, 2013). Articles were organized into 29 topic trees by day. The tree depths varied from 6 to 9, the total node numbers changed from 1,919 to 5,419, and the node number of the first level ranged from 554 to 1,303.

Three metrics were introduced to assess the smoothness between adjacent tree cuts.

Tree mapping ($\boldsymbol{S}_{map}$): This metric is derived from the proposed global tree cut energy function, $\boldsymbol{S}_{map} = \mathbf{E}_1 + \mathbf{E}_3$, where $\mathbf{E}_1$ and $\mathbf{E}_3$ are defined in Eq. (3). The smaller the tree mapping value, the higher the smoothness between two adjacent trees.

Normalized Mutual Information (NMI) ($\boldsymbol{S}_{NMI}$): The NMI metric is the mutual information between cluster assignments and a pre-existing labeling. This metric was then adopted to assess the similarity between adjacent tree cuts. The larger the NMI value, the higher the smoothness between two adjacent trees. In our experiment, the Hungarian algorithm [30] was employed to find the optimal match between the document sets of the two tree cuts.

Tree distance ($\boldsymbol{S}_{dist}$): This metric measures the tree cut difference by aggregating the tree distance difference between two related cut nodes of the adjacent trees. This metric is defined as $S_{Dist} = \frac{1}{\lambda}\log(p_{dist}(T^t|T^{t-1}))$, where $\log p_{dist}(T^t|T^{t-1})$ is:

$$\log p_{dist}(T^t|T^{t-1}) \triangleq -\lambda E_{\substack{r,s\in leaves(T^t)\\ r\neq s}} (d_{T^t}(r,s) - d_{T^{t-1}}(r,s))^2, \qquad (10)$$

where $d_T(r,s)$ is the tree distance between nodes r and s. λ is for balancing smoothness and tree likelihood. The smaller the tree distance value, the higher the smoothness between two adjacent trees.

The fitness measures how satisfactorily the topics on the tree cut describe the topic distribution within a topic tree based on user interest. According to Eq. (2), the baseline tree cut of each tree can better represent the topic and thus has the better fitness. Thus, for each topic, one major goal of the evolutionary tree cut algorithm is to generate a tree cut that is as similar as possible to the baseline tree cut. Accordingly, the NMI metric ($\boldsymbol{F}_{NMI}$, content similarity) and tree distance metric ($\boldsymbol{F}_{dist}$, structure similarity) were adopted to measure the fitness of the evolutionary tree cut, namely, to compare the content and structure similarity between the evolutionary tree cut and the baseline tree cut. Typically, the closer the value of $\boldsymbol{F}_{NMI}$ is to 1, the better the fitness; the smaller the $\boldsymbol{F}_{dist}$, the better the fitness.

In our experiments, focus nodes were randomly selected to avoid any biased conditions. To further eliminate the randomness caused by the focus node selection, we randomly selected the given number of focus nodes 50 times and ran the experiment 50 times. Results were computed by averaging the 50 trials.

We compared the overall smoothness and fitness with the baseline. As shown in Table 1, our method not only maintains good fitness values under the $\boldsymbol{F}_{NMI}$ metric (close to 1) and the $\boldsymbol{F}_{dist}$ metric (less than 1), but also generates a smoother structure compared to that of the baseline. In addition to performing very well under the metric of $\boldsymbol{S}_{map}$, our algorithm also achieved a comparable performance under the metrics of $\boldsymbol{S}_{NMI}$ and $\boldsymbol{S}_{dist}$.

Table 1. Evaluation of smoothness. $f_r := \frac{m_o - m_b}{m_b} * 100\%$, m_b is the metric value of the baseline, and m_o is the metric value of our method.

	$f_r(\boldsymbol{S}_{NMI})(\%)$			$f_r(\boldsymbol{S}_{dist})(\%)$			$f_r(\boldsymbol{S}_{map})(\%)$			$\boldsymbol{F}_{NMI}$			$\boldsymbol{F}_{dist}$		
	1 focus	2 foci	3 foci	1 focus	2 foci	3 foci	1 focus	2 foci	3 foci	1 focus	2 foci	3 foci	1 focus	2 foci	3 foci
Dataset A	10.27	15.41	14.27	-27.06	-23.91	-19.87	-30.84	-31.63	-34.65	0.9602	0.9689	0.9752	0.2965	0.2677	0.2512
Dataset B	17.95	22.01	29.55	-20.57	-25.29	-25.64	-30.46	-30.12	-29.57	0.9083	0.9138	0.9182	0.4815	0.5083	0.5124
Dataset C	24.48	25.77	17.71	-37.07	-39.56	-28.82	-48.14	-39.99	-39.33	0.9207	0.9116	0.9075	0.7238	0.8358	0.8303

7.2 User Study

To evaluate the effectiveness of the proposed stable layout method, we designed and conducted a comparison study on Amazon Mechanical Turk (AMT). Our study compared the stability and readability of our stable layout with the baseline method, the optimization algorithm that aims to minimize edge crossings.

7.2.1 Data

We generated 30 initial layouts that were sampled from the experiment data used in the quantitative evaluation. The average number of edge crossings of the 30 initial layouts was 44 (min = 4, max = 149, s.d. = 41.1). The numbers of time points, topics, and edges for each initial layout ranged from 11 to 21, 74 to 223, and 98 to 249, respectively. For each initial layout, one random edit was made to the back-end tree cuts. Accordingly, a new set of tree cuts was generated by the incremental evolutionary tree cut algorithm. Afterwards, the stable and optimization layout algorithms were applied to the newly generated tree cuts. The stable layout that aimed to preserve the mental map and the optimized layout that aimed to minimize the edge crossings were obtained for our user study tasks.

7.2.2 Tasks and Setup

Three tasks (from simple to difficult) were designed to evaluate participants' ability to track topics during exploration and identifying the stability preserving capability of the two layout methods.

Similarity: Given an initial layout, this task aims to find a more similar layout to the initial one from the two layouts generated by our method and the baseline method. In the user study setup, 30 instances were shown one by one to each participant. For each instance, all three layouts (initial, stable, and optimized) were displayed side-by-side as three separate images. The initial layout was labeled as A, whereas the corresponding stable and optimized layouts were randomly labeled as B and C, respectively, to avoid potential biases. Each participant was asked to compare them and decide whether B or C was more similar to A. The decisions were recorded for further analysis.

Visual clutter: This task compares the visual clutter between the stable layout and the optimized layout. In the setup, 30 instances were used. For each instance, the stable and optimized layouts were displayed side-by-side in random order. Each participant was required to choose which one looks better than the other in terms of visual clutter and rate the advantage (1 = a little better, 5 = a lot better). The decisions and rates were recorded for further analysis.

Topic tracking: This task compares the performance of the stable and the optimized layout methods in tracking corresponding topics between the initial and new layouts. This is the most difficult task among the three. 15 instances were used in this task.There are 2 layout pairs for each instance. Each pair contains the initial layout and one new layout that was either the stable or optimized layout. The 30 total pairs were shuffled and shown one by one to each participant. In each pair, one topic was marked in the initial layout. With our incremental tree cut algorithm, the marked topic might 1) still exist; 2) be split into multiple nodes; 3) or merge with other nodes in the new layout. The participant is required to find the corresponding topic in the new layout. To simplify the task implementation and avoid ambiguity, we marked four candidates in the new layout for the participant (Fig 10). The completion time and decision were recorded for further analysis.

Considering the simple perceptual nature of our tasks, no qualification tests were required to complete our Human Intelligence Tasks (HITs). The only requirement for all the participants was the previous HIT approval rate of 95% or higher. To test and avoid irresponsible workers, two testing instances were placed in each task. For each task, over 300 workers were recruited, who were each paid $0.50, when the

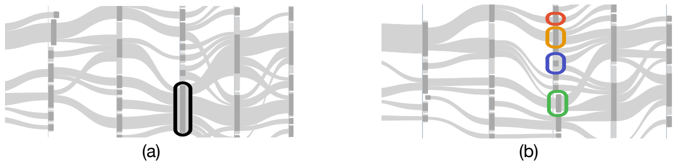

Figure 10. Tracking task example: (a) a topic is marked in the original layout; (b) four candidates of the corresponding topic are marked for the participant to choose.

task was finished. In addition, to ensure fairness, participants in one task could not take part in any of the other tasks.

After collecting the data, we rejected 24 participants because they provided incorrect answers to the testing instances, or provided answers before the layout images were completely loaded.

7.2.3 Results and Discussions

Similarity: Answers from 298 participants were accepted. Fig. 11 shows the final results. We observed a mean of 79.4% (min = 0.644, max = 0.932, s.d. = 0.014, *p-value* $\ll$ 0.001) among the participants who chose the stable layouts over the corresponding optimized layouts. The observed results matched our expectation that the stable layout outperformed (statistically significant) the optimized method in preserving the mental map.

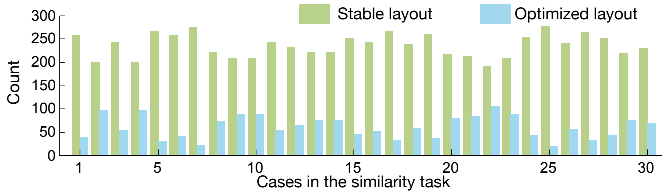

Figure 11. Number of participants voting for stable or optimized layouts of the similarity task.

Visual clutter: Answers from 299 participants were accepted. The final results are displayed in Fig. 12. We observed a mean score of 2.76 (s.d. = 1.49) for participants who chose the stable layouts, whereas a mean score of 2.77 (s.d. = 1.52) was observed for those who chose the optimized layouts. Statistically, the participants did not show significant preferences to either the stable layouts or the corresponding optimized layouts (*p-value*=0.344). We also compared the number of edge crossings in the stable layouts and corresponding optimized layouts. The results show that the stable layouts, on average, possessed 2.6 extra edge crossings, compared with their corresponding optimized versions. Although people may have different standards of visual clutter, this could be considered as one major reason participants did not particularly prefer the optimized layouts or the stable layouts.

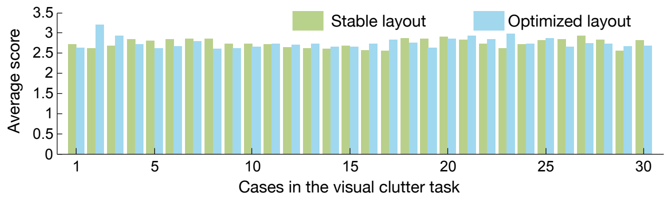

Figure 12. Average score of the visual clutter task.

Topic tracking: Answers from 300 participants were accepted. For each instance, a participant took an average of 10.2 seconds (min = 1.81, max = 48.3, s.d. = 8.44) to decide on the stable layouts, whereas they took 13.6 seconds (min=1.86, max =59.1 s.d.=11.2) to decide on the optimized layouts. The final results are illustrated in Fig. 13.

We observed that the participants used less time to achieve a higher accuracy rate for the results generated by the stable layout method. Overall, the differences between the completion times as well as the accuracy rates are both statistically significant (*p-value*=0.000163 and *p-value*=1.2e-282). In Fig. 13, we also observed two outliers, #2 and #7. Thus we closely examined the related layout results and attempted to understand the reason. For instance #2, the low accuracy rates for both the stable and optimized layouts were caused by a misleading topic, which is extremely similar to the topic that the participants were asked to track. Although the stable algorithm maintained the corresponding topic close to its position in the initial layout, participants were still confused by the misleading topic and provided wrong answers. For instance #7, both the stable and the optimized layouts placed the corresponding topic at the same place in the initial layout. Thus, the participants could easily find the correct answers in both layouts, which resulted in high accuracy rates.

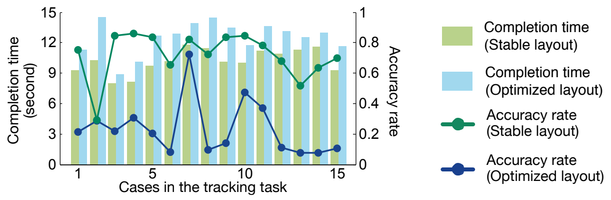

Figure 13. Average task completion time and accuracy rate of the topic tracking task.

7.3 Case Study

From the interviews and discussions with our domain experts, we derived several usage scenarios. In this section, we take the Prism scandal as an example to demonstrate how *RoseRiver* helps experts explore and gain insights into their datasets.

The statistics for the Prism dataset, which was provided by a professor in social media and communication, are summarized in Sec. 7.1. The professor participated in our case study and attempted to explore a variety of topics and their evolution patterns in the dataset.

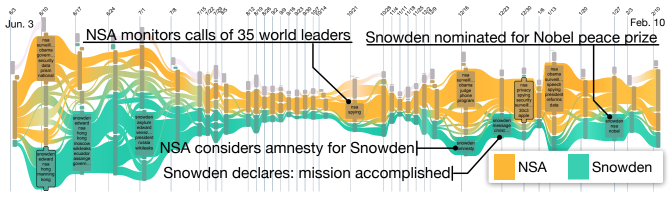

Figure 14. Overview of evolving hierarchical topics in the Prism dataset (Jun.3, 2013 to Feb 9, 2014).

Without any user input, a high-level overview (Fig. 14) was automatically generated and presented to the professor. Our system used the mean-shift algorithm [9] to cluster the topics at the first level because it was the most abstract level and can represent the topic tree very well. For each cluster, we chose the topic closest to the cluster center as one focus topic input to our tree cut algorithm. Two major topics, namely, "NSA" (orange) and "Snowden" (cyan), were detected. Gray topics were not related to either of the focus topics. The "NSA" (orange) topic mostly discussed the NSA spying activities and related privacy/security subjects. The "Snowden" (cyan) topic focused more on tracking and reporting news related to the whistleblower Snowden.

The "Snowden" (cyan) topic was stronger than the "NSA" (orange) topic in the first month, but it shrank quickly over time. The "NSA" (orange) remained stable all the time. According to the professor, this phenomenon can be interpreted by "issue-oriented reporting" and "event-oriented reporting" theories [31]. The professor explained that, although these two topics had the same origin, "NSA" (orange) topic contained a classic media issue about privacy and security, which led to a stable and strong tail in its life cycle. On the other hand, the "Snowden" (cyan) topic was more event-oriented, which would disappear quickly if no new events occurred to reactivate it.

Observing that the data volume is large and the relationships are complex in the first few months, the professor suggested examining that time period with a smaller time step. Thus, we zoomed into June and August and changed the time step to two days (Fig. 1(a)).

In Fig. 1(a), four major topics were encoded by different colors: "Obama" (red), "Prism" (purple), "Asylum" (green), and "EU allies" (blue). The "Obama" (red) topic included all news and tweets related to the actions that Obama and the US government took against the Prism scandal. This topic frequently interacted with the "Prism" topic, which focused on revealing and discussing new information about the scandal. The professor observed pink topics in many time points, which indicated that these two topics often merged together.

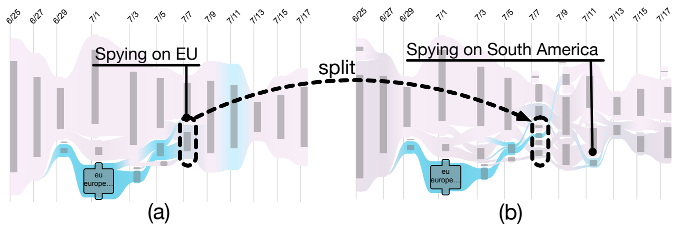

Figure 15. (a) Distribution of the "EU Allies" topic in the dataset. (b) New tree cut result after the node split.

The "EU allies" (blue) topic concerned the NSA spying on its European allies and related discussions. Judging from its size and connections to the other topics (Fig. 1(a)**A**), it was sudden and short-lived. The professor found this pattern surprising because he believed this event was a "game changer" and should have triggered a more profound discussion. From this visualization, however, it appeared more event-oriented than issue-oriented. Thus, we chose this topic as the sole focus topic and re-ran our evolutionary tree cut algorithm. Fig 15(a) shows the distribution of this topic in the dataset. There were only a few time points highly related to this topic, which makes it a short-lived topic. The professor still wanted to see more details about this topic. We manually split one of the related topics. Thus, *RoseRiver* inputted this split operation into the optimization process and re-generated a new set of tree cuts. As shown in Fig. 15(b), more detailed topics appeared. In particular, the professor noticed that a small blue topic was automatically extracted from the big topic on Jul. 11. By examining the content, the professor found these topics were divided into two parts: "NSA bugged EU offices"(Jun. 29 - Jul. 7) and "NSA Spying on Latin American Countries" (Jul. 11). However, neither of them triggered new discussions to keep this topic active.

Because he played a key role in the scandal, Snowden was another study subject of the professor. In Fig. 1(a), the green stripes depict Snowden's asylum journey. The topic of Snowden first appeared on Jun. 7. The professor was curious about the difference between the content in tweets and news articles of the topic. Thus we extracted the most frequent keywords from these two sources and displayed them around this topic as word clouds (Fig. 1(b)). Several interesting patterns were immediately pointed out by the professor. For example, he observed that the prominent keywords were clearly different between news and tweets. In the word cloud generated from news, frequent keywords were "secret," "booz allen," "leaker," and "contractor," indicating news media was busy exposing details in this event, such as "Booz Allen falls after employee intel leak." In contrast, the prominent keywords from tweets were "tcot," "privacy," and "freedom." The professor commented, "you see, their focuses are so different. No matter what is reported in the news, people mainly see and care what is valuable to themselves." In addition, the professor discovered that two words, "hero" and "traitor," appeared on both Twitter and the news. In the word cloud for tweets, "hero" clearly out-weighed "traitor," such as "#Hero Edward Snowden: the whistleblower behind revelations of NSA surveillance via." However, in the word cloud for news articles, they are similar in size. The professor clicked the keyword "hero" to examine the detailed news articles and found that most of them were discussions, such as "Edward Snowden: a hero or traitor?" He commented, "This probably explains why they have the same size. Journalists prefer igniting discussions rather than making judgments." On Jun. 9, the topic immediately split into two topics. One (Fig. 1(a)**B**) was related to Snowden but not asylum, such as "Booz Allen fires leaker Snowden" and "Boehner: Edward Snowden is a traitor," which quickly disappeared. The other (Fig. 1(a)**C**) continued to follow his asylum journey, which ended around Aug. 2 when Snowden was finally granted temporary asylum in Russia. Afterwards, it merged into a gray topic (Fig. 1(a)**D**), which was connected to the "Obama" (red) topic. The professor was curious about what caused this transition. We then split the gray topic into smaller topics to explore in greater detail. Fig. 1(c) shows the result generated by our incremental evolutionary tree cut algorithm. On Aug 8, a yellow topic (Fig. 1(c)**E**) appeared and connected the green and red topics. The yellow topic concerned "Obama urged to cancel Putin meeting over Snowden asylum," which clearly illustrated the major reason for the transition. To the expert's surprise, there was another yellow topic "Obama urges Putin to look forward, not back" appeared on Aug. 12 (Fig. 1(c)**F**). The professor commented that it was an unexpected pattern to him and he would like to investigate what caused this repetition in his study.

7.4 Initial Expert Feedback

We have worked closely with a group of experts, including one public relations manager, one professor in political science, and two professors in media and communications. When developing *RoseRiver*, we continuously collected their feedback to improve our design. After several iterations, *RoseRiver* has been well received by the experts. They agreed this system can greatly reduce their workload in performing tedious text corpus analysis. In this section, we summarize their feedback into three categories.

7.4.1 Evolutionary Tree Cut

All the experts experienced some difficulty in finding interesting topics by using *TextFlow*. Thus, they were eager for a more effective way to manage hierarchical evolving topics. When *RoseRiver* was first presented to them with the evolutionary tree cut feature, the experts immediately agreed it was a very helpful function. They commented that the focus changing feature made the data "enjoyable to explore" and it allowed them to "try all kinds of ideas they could think of." In addition, we also asked the experts to compare *RoseRiver* with ThemeRiver-based visual analytic systems such as TIARA [22] and Hierarchical-Topics [11]. Overall, *RoseRiver* was most favored by the experts. For example, one professor in media and communications said, "Purely individual topics are less useful for analyzing competition among issues. With *RoseRiver*, I can see different trees without losing the forest." Another professor in political science commented that the splitting/merging patterns are particularly useful in his public administration research, such as tracking the influence between different topics.

7.4.2 Mental Map Preservation

We provided two systems to our target users. One system employs the evolutionary tree cut algorithm and the optimized layout algorithm (baseline). The other is our system with an incremental evolutionary tree cut algorithm and the stable layout algorithm. After trying both systems to analyze their own data, the experts reported a huge difference in the user experience to the two systems. All the experts agreed that our system outperformed the baseline system in providing more meaningful context for further investigation. For example, the public relations manager commented that "the exploration suddenly becomes smooth and I can easily find the topics of interest that were identified before in the new visualization."

In addition to the mental map preservation, they also appreciated the intuitive interactions. The incremental evolutionary tree cut algorithm was initially designed to keep the stability between old and new tree cut results. However, beyond expectation, we found that the experts were more interested in the changed part in the new tree cut result. It turned out our experts used the incremental evolutionary tree cut algorithm as a "recommendation system." The professor in environmental politics pointed out, "It looks like the system is guiding me to check these new topics as well (pointing to some topics)! It is so easy and useful for a computer dummy like me."

7.4.3 Improvements

The experts also made several suggestions to improve *RoseRiver*. Three of them would like to be able edit the evolutionary topic trees through our system interface. We had heard many comments like "it will make more sense to me if these two topics are combined." The experts believed the results could be greatly improved if they can use their knowledge to refine the topic trees. The public relations manager would like us to extend this system to support streaming data. She commented that "We have to follow the news and Twitter data every day and take actions promptly, so it would be a huge help if your system can help analyze my data in a streaming manner." This comment indicates an important direction for future research, namely, to support text streams. The key is to extend our tree cut propagation algorithm to process topic trees in a streaming manner. One possible solution is to use the previous tree cut results as a constraint when running the optimization algorithm on newly arrived data.

8 Conclusion and Discussion

We have presented a visual topic analytics system, *RoseRiver*, to help users better understand the hierarchical topic evolution at different levels of granularity. Key aspects of our system include an incremental evolutionary tree cut algorithm and a stable graph layout algorithm to preserve the mental map. By preserving a stable context, our system allows users to progressively explore and analyze the complex evolution patterns of hierarchical topics.

Although *RoseRiver* performs well when analyzing the evolution of hierarchical topics, it still has some limitations. First, the quality of tree cut results highly depend on the quality of the input evolutionary topic trees. Ill-structured trees may result in some meaningless tree cuts, which makes it much more difficult for users to understand. To solve this problem, it is desirable to study how to integrate a user's domain knowledge into our system and allow him/her to interactively refine the input topic trees. The key is to seamlessly integrate the evolutionary tree clustering model with the incremental tree cut algorithm and interactive visualization. This is an interesting research topic to pursue in the future. Second, even if the input topic trees are well-structured, not all tree cuts are meaningful. The quality of the tree cut result depends on the chosen focus topics and their relationships with each other. When conducting the case studies, we observed that if the user has no prior knowledge about the data, it usually took four or five trials to obtain a meaningful result. We also observed that through the failed attempts, the user became more familiar with the dataset, which led to a higher success rate of finding interesting focus topics. One intuitive solution to reduce the number of failed trials is ranking all topics based on different criteria and making recommendations accordingly. Third, the incremental evolutionary tree cut algorithm is more effective for editing the existing tree cut results. When the user change the focus topics completely, our system will reset the $\{1\}$ and $\{0\}$ topic sets. Thus, our algorithm may fail to preserve the stability between the old and new tree cut results. Our case studies showed that the experts experienced some difficulty in mentally connecting the new layout with the old one for some focus changes. However, they also commented that it was acceptable since focus changes usually meant the analysis tasks were changed and they did not need to mentally connect the new layout with the old one in most cases. Nonetheless, it is a more profound issue and worth thoroughly investigating in the future. Fourth, but not least, tree structures are only partially encoded in our system. In our visual design, we use horizontal offsets to encode the tree depth, which can reveal the relative levels between nodes. However, in some cases, users want to examine each tree structure and get a complete overview of evolving topic trees. Thus, we plan to integrate the ability of tree navigation into *RoseRiver* and provide a more comprehensive hierarchical topic exploration system.

Acknowledgments

We gratefully thank all the domain experts for their contributions to this system. We would also like to thank Xiting Wang and Jialun Yin for their help during the quantitative evaluation and the anonymous reviewers for their valuable and constructive comments.

REFERENCES

[1] Mosek package. `http://www.mosek.com`, March 2014.

[2] D. Archambault and H. C. Purchase. Mental map preservation helps user orientation in dynamic graphs. In *Graph Drawing*, pages 475–486, 2012.

[3] D. Archambault, H. C. Purchase, and B. Pinaud. Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *IEEE Transactions on Visualization and Computer Graphics*, 17(4):539–552, 2011.

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

[5] C. Blundell, Y. W. Teh, and K. A. Heller. Bayesian rose trees. In *UAI*, pages 65–72, 2010.

[6] Y. Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *ICCV*, volume 1, pages 105–112, 2001.

[7] S. K. Card and D. Nation. Degree-of-interest trees: A component of an attention-reactive user interface. In *AVI*, pages 231–245, 2002.

[8] C. Collberg, S. Kobourov, J. Nagra, J. Pitts, and K. Wampler. A system for graph-based visualization of the evolution of software. In *ACM symposium on Software visualization*, pages 77–86, 2003.

[9] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[10] W. Cui, S. Liu, L. Tan, C. Shi, Y. Song, Z. Gao, H. Qu, and X. Tong. Textflow: Towards better understanding of evolving topics in text. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2412–2421, 2011.

[11] W. Dou, L. Yu, X. Wang, Z. Ma, and W. Ribarsky. Hierarchicaltopics: Visually exploring large text collections using topic hierarchies. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2002–2011, 2013.

[12] G. W. Furnas. Generalized fisheye views. In *CHI*, pages 16–23, 1986.

[13] E. Hatcher, O. Gospodnetic, and M. McCandless. Lucene in action, 2004.

[14] S. Havre, E. G. Hetzler, P. Whitney, and L. T. Nowell. Themeriver: visualizing thematic changes in large document collections. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):9–20, 2002.

[15] M. Jnger, E. Lee, P. Mutzel, and T. Odenthal. A polyhedral approach to the multi-layer crossing minimization problem. In G. DiBattista, editor, *Graph Drawing*, volume 1353 of *Lecture Notes in Computer Science*, pages 13–24. Springer Berlin Heidelberg, 1997.

[16] J. Leskovec, L. Backstrom, and J. M. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD*, pages 497–506, 2009.

[17] Y. Li, J. Sun, and H.-Y. Shum. Video object cut and paste. *ACM Transactions on Graphics*, 24(3):595–600, 2005.

[18] S. Liu, W. Cui, Y. Wu, and M. Liu. A survey on information visualization: recent advances and challenges. *The Visual Computer*, pages 1–21, 2014.

[19] S. Liu, X. Wang, J. Chen, J. Zhu, and B. Guo. Topicpanorama: a full picture of relevant topics. *To appear in IEEE Transactions on Visualization and Computer Graphics*, 20(12), 2014.

[20] S. Liu, Y. Wu, E. Wei, M. Liu, and Y. Liu. Storyflow: Tracking the evolution of stories. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2436–2445, 2013.

[21] S. Liu, M. X. Zhou, S. Pan, W. Qian, W. Cai, and X. Lian. Interactive, topic-based visual text summarization and analysis. In *CIKM*, pages 543–552, 2009.

[22] S. Liu, M. X. Zhou, S. Pan, Y. Song, W. Qian, W. Cai, and X. Lian. Tiara: Interactive, topic-based visual text summarization and analysis. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(2):25:1–25:28, 2012.

[23] X. Liu, Y. Song, S. Liu, and H. Wang. Automatic taxonomy construction from keywords. In *KDD*, pages 1433–1441, 2012.

[24] D. Luo, J. Yang, M. Krstajic, W. Ribarsky, and D. A. Keim. Eventriver: visually exploring text collections with temporal references. *IEEE Transactions on Visualization and Computer Graphics*, 18(1):93–105, 2012.

[25] P. McLachlan, T. Munzner, E. Koutsofios, and S. North. Liverac: interactive visual exploration of system management time-series data. In *CHI*, pages 1483–1492, 2008.

[26] M. Monroe, R. Lan, H. Lee, C. Plaisant, and B. Shneiderman. Temporal event sequence simplification. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2227–2236, 2013.

[27] R. Munroe. Movie narrative charts. http://xkcd.com/657/.

[28] T. Munzner. A nested model for visualization design and validation. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):921–928, 2009.

[29] M. Ogawa and K.-L. Ma. Software evolution storylines. In *Proceedings of the 5th international symposium on Software visualization*, pages 35–42, 2010.

[30] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.

[31] J. W. Rhee. Strategy and issue frames in election campaign coverage: A social cognitive account of framing effects. *Journal of Communication*, 47(3):26–48, 1997.

[32] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[33] K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man and Cybernetics*, 11(2):109–125, 1981.

[34] G.-D. Sun, Y.-C. Wu, R.-H. Liang, and S.-X. Liu. A survey of visual analytics techniques and applications: State-of-the-art research and future challenges. *Journal of Computer Science and Technology*, 28(5):852–867, 2013.

[35] Y. Tanahashi and K.-L. Ma. Design considerations for optimizing storyline visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2679–2688, 2012.

[36] F. van Ham and A. Perer. search, show context, expand on demand: Supporting large graph exploration with degree-of-interest. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):953–960, 2009.

[37] X. Wang, S. Liu, Y. Song, and B. Guo. Mining evolutionary multi-branch trees from text streams. In *KDD*, pages 1433–1441, 2013.

[38] K. Wongsuphasawat and D. Gotz. Exploring flow, factors, and outcomes of temporal event sequences with the outflow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18:2659–2668, 2012.

[39] K. Wongsuphasawat, J. A. Guerra Gómez, C. Plaisant, T. D. Wang, M. Taieb-Maimon, and B. Shneiderman. LifeFlow: visualizing an overview of event sequences. In *CHI*, pages 1747–1756, 2011.

[40] D. D. Woods. Visual momentum: a concept to improve the cognitive coupling of person and computer. *International Journal of Man-Machine Studies*, 21(3):229–244, 1984.

[41] Y. Wu, S. Liu, K. Yan, M. Liu, and F. Wu. OpinionFlow: Visual analysis of opinion diffusion on social media. *To appear in IEEE Transactions on Visualization and Computer Graphics*, 20(12), 2014.

[42] P. Xu, Y. Wu, E. Wei, T.-Q. Peng, S. Liu, J. J. H. Zhu, and H. Qu. Visual analysis of topic competition on social media. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2012–2021, 2013.

[43] D. Zhang, C. Zhai, J. Han, A. N. Srivastava, and N. C. Oza. Topic modeling for olap on multidimensional text databases: topic cube and its applications. *Statistical Analysis and Data Mining*, 2(5–6):378–395, 2009.