Author manuscript, published in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 259-268, 2015.
DOI: 10.1145/2702123.2702419

MatrixWave: Visual Comparison of Event Sequence Data

Jian Zhao[1] **Zhicheng Liu**[2] **Mira Dontcheva**[2] **Aaron Hertzmann**[2] **Alan Wilson**[2]
[1]University of Toronto [2]Adobe Research

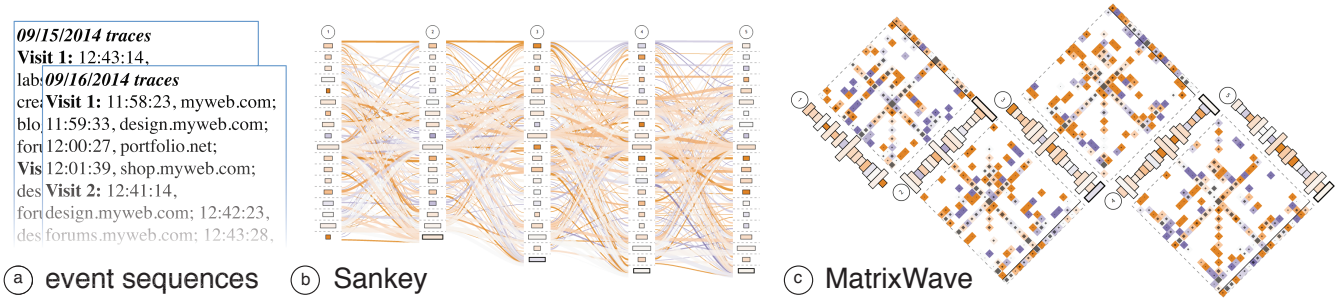

Figure 1. We explore visualization designs for comparison analysis of two web clickstream datasets (a). Node-link diagrams, like the Sankey diagram (b), use a familiar node-link paradigm, but lead to many overlapping edges. In contrast, MatrixWave displays links as with matrices (c) and arranges step-to-step transitions in a zig-zag pattern. Traffic volume is encoded with size, and differences between the two datasets are encoded with color.

ABSTRACT

Event sequence data analysis is common in many domains, including web and software development, transportation, and medical care. Few have investigated visualization techniques for comparative analysis of multiple event sequence datasets. Grounded in the real-world characteristics of web clickstream data, we explore visualization techniques for comparison of two clickstream datasets collected on different days or from users with different demographics. Through iterative design with web analysts, we designed MatrixWave, a matrix-based representation that allows analysts to get an overview of differences in traffic patterns and interactively explore paths through the website. We use color to encode differences and size to offer context over traffic volume. User feedback on MatrixWave is positive. Our study participants made fewer errors with MatrixWave and preferred it over the more familiar Sankey diagram.

Author Keywords

Event sequences; visual comparison; Sankey diagram; matrix representation; information visualization.

ACM Classification Keywords

H.5.2. Information Interfaces and Presentation (e.g. HCI): User Interfaces

INTRODUCTION

Analysts often seek to understand event sequence data, that is, multiple series of timestamped events. Event sequence data arises in many applications. For example, websites log how users navigate their pages, airlines track events during airplane flights, and hospitals record when patients transfer from one part of the hospital to another. A key analysis task is *comparison* between different time periods and different populations. For example, a website analyst asks numerous questions about how users navigate their site. Is traffic increasing or decreasing? Are this week's visitors leaving the site faster than last week's visitors? Are visitors from North America visiting the same parts of the website as visitors from Europe? While several visualization techniques have been designed for event sequence data, such as the Sankey diagram [25] and the LifeFlow [31] and Outflow [30] systems, little attention has been paid to supporting visual comparisons for this type of data.

In this work, we explore visualization designs for *comparative analysis* of two related event sequence datasets. We perform this exploration in the context of web clickstream data, but we believe our findings generalize to other domains. As we began our exploration, we found that visualizing even a single clickstream dataset is not well supported by existing techniques, as clickstream data includes many events and many transitions resulting in complex and unreadable visualizations (Figure 1-b). To handle visualizing real-world clickstream data, we propose the MatrixWave representation. MatrixWave displays a transition matrix [3] for each successive step in the sequence, rotated and concatenated side-by-side in a zig-zag manner (Figure 1-c). This design directly exposes traffic statistics for individual nodes and for step-to-step transitions; moreover, individual navigation paths through the site can be isolated as zig-zag paths through the sequence of matrices. To support comparative analysis of event sequences, we design additional visual encodings using color and size. Using our design, an analyst can get an overall picture of website traffic from the basic visualization, can identify popular nodes and transitions from histogram statistics, and can see how these quantities have changed over time or between user segments. MatrixWave also incorporates flexible interactions, such as filtering, searching, and node reordering, to aid users with the exploration of data.

CHI 2015, April 18–23 2015, Seoul, Republic of Korea.

http://dx.doi.org/10.1145/2702123.2702419

We grounded our design in interviews with expert web analysts and iteratively collected feedback throughout the design process. Additionally, we evaluated how well users perform with MatrixWave and the Sankey diagram from comparison tasks. Although the Sankey diagram is more familiar and MatrixWave requires learning a new technique, the study participants made fewer errors and preferred the MatrixWave representation over the Sankey diagram. Qualitative feedback points to the need for better ways to filter the data and further exploration into layout.

RELATED WORK

Visualizations of clickstream data

Clickstream visualization has received considerable attention. ClickViz [5] and WebQuilt [13] introduced directed graphs for clickstream visualization, but these approaches can only handle dozens of paths. In order to handle larger datasets, some authors have proposed clustering paths [7, 14, 29] and showing summary statistics [17]. These approaches provide a high-level overview of clickstream patterns, but do not allow fine-scale analysis of individual paths. Moreover, the actual set of behaviors may not be represented well by clusters. Commercial systems, such as Google Analytics, use Sankey diagrams to show pages and links with the most traffic, but offer little assistance with comparison.

Visualization of event sequences

Perhaps the most basic approach to event sequence visualization is to display the raw sequences, as done by Lifelines [24, 28], CloudLines [15], and TimeSlice [33]. These methods can provide considerable detail to individual traces, but do not scale to clickstream data, which typically involves thousands of sequences. In order to reduce dataset size, LifeFlow [31] and EventFlow [19] consolidate common subsequences, resulting in a tree-like representation. These representations scale exponentially in the number of event types, and thus are impractical for clickstream data, where each event is typically one of hundreds of webpages.

Several authors have proposed consolidating events into a graph. A state transition diagram [27] is a general graph of state nodes that directly shows all routes between pages. But, this approach does not show step information and does not scale well to highly-connected graphs. In some cases, the scaling issue can be addressed by discovering motifs [18, 23], or conducting interactive queries over event types, event intervals, etc. [10, 20]. The Sankey diagram [25] is a form of graph arranged to show step-by-step transitions; we discuss the Sankey more extensively later in the paper. A number of variants and extensions to the Sankey have also been developed [30].

Two previous methods use sequences of matrices, similar to MatrixWave. MatrixFlow [22] visualizes histories as a sequence of matrices. Though superficially similar to MatrixWave, MatrixFlow does not describe paths through states, as each matrix is a separate, independent visualization, and each matrix is a symmetric correlation matrix rather than a transition matrix. Closest to our own work is GeneaQuilts [4], which visualizes genealogy by alternating individuals with pairs of matrices. This work is specialized to genealogy, in which each child has exactly two parents. The GeneaQuilts graph requires twice as many matrices for the same number of steps as MatrixWave, in order to completely specify ancestry. Although MatrixWave would not be able to precisely show ancestry, it is designed for much denser transition relations where each event may follow many other events, and where there are many more possible events.

Visual data comparison

Gleicher et al. [9] classify visual comparison techniques into three categories: juxtaposition, superposition (e.g., [26]), and explicit encoding (e.g., [11]). Another common approach to reveal differences between data is to use animation (e.g., [2]). Juxtaposed views enable a side-by-side comparison of data, often complemented with interactions highlighting the matches or explicit drawings showing the connections [21, 12, 6, 32]. Superposed views allow an in-place comparison by overlaying multiple objects in one visualization [26, 16]. Explicit encoding techniques compute the differences between related elements and visualize these derived values [11]. Finally, animated views interpolate two visual representations to preserve a viewer's mental map [2]. These prior works on visual comparison primarily focus on trees and graphs, and have not addressed the visualization and comparison of changes between two event sequences. Nevertheless, they provide valuable results to ground our design choices. We employ superposition to combine two related event sequence datasets and use explicit encoding for comparison tasks.

UNDERSTANDING CLICKSTREAM ANALYSIS

We use web clickstream data to ground our investigation in real-world problems. Modern commercial websites track thousands of distinct clickstream sequences each single day. A sequence consists of ordered events, each assigned a step number. Each event is an interaction with the website, including a click on a link, a click on a menu, or filling out a form and pressing return. These sequences are highly variable. Some include only one or two events, while others can include tens or hundreds of unique events.

To better understand the web analysis domain, we interviewed expert analysts who work with clickstream data at large IT companies. We conducted multiple interviews and collected feedback throughout the design process. During the interviews, we presented the prototypes, discussed alternatives for visual encodings and interactions, and collected feedback from the experts to guide our next steps. Although we developed MatrixWave in the context of comparing web clickstream data, the visualization and its interactive features can be generalized to other types of event sequence data.

To ground our design in real work, we distilled the following analysis tasks from our interviews:

Traffic volume: One of the most common analysis tasks is assessing the number of visitors to a site and tracking which pages get the most visitors. Analysts usually compare traffic volume from a given day to volume from the previous day and volume from the same day on the previous week. Volume

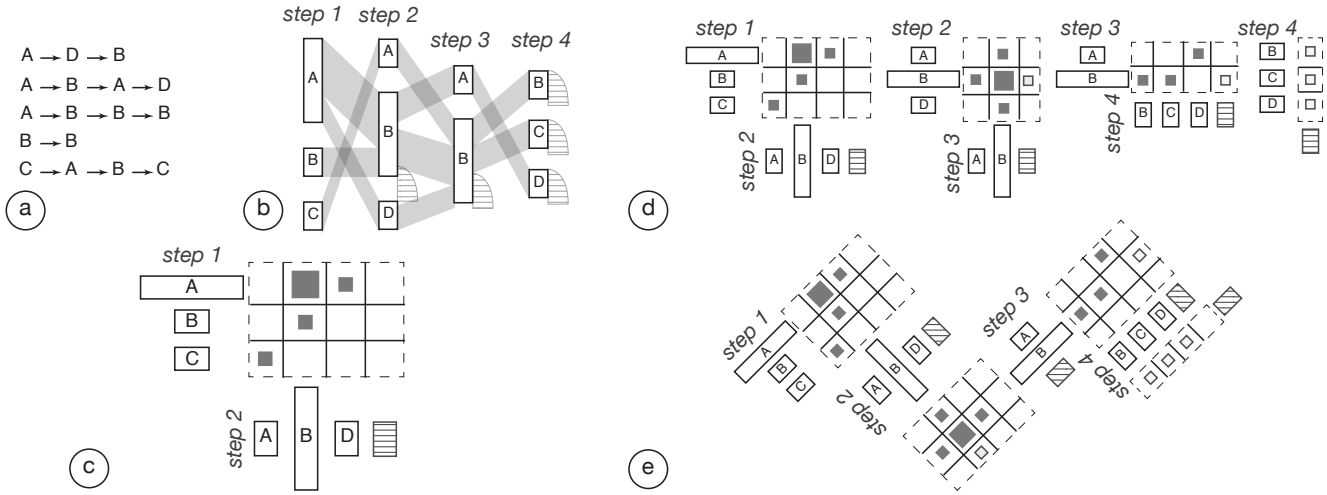

Figure 2. (a) Clickstream data comprises sequences of web pages, each one describing a unique visit by a visitor to a website. (b) The conventional visualization for clickstreams is the Sankey diagram, in which each column of bars represents a step number. But Sankey diagrams can become unreadable due to edge crossings between links. (c) We propose to replace the links with a transition matrix, in which each element in the grid represents the link from one node to the next; here, the size of the gray square indicates the traffic volume in that link. (d) Matrices can be sequenced together. However the notion of a path through the website is lost. (e) To enable path visualization and eliminate duplicating nodes, we chose a zig-zag layout. Drop-off (exit) nodes are filled in with a line pattern.

comparison is also common across steps in the sequence. Volume comparison across steps tells analysts how quickly visitors leave the site.

Entry into the website: Analysts often seek to understand the effect of marketing campaigns: are they bringing more people to the website, and what are these people doing? For this task, analysts break down traffic by entry point (email click, search engine, banner ad, etc.) and study where new visitors go next as compared to other types of visitors. Close inspection of the first one or two steps tells analysts how many people continue deeper into the site.

Exits from the website: Analysts also seek to understand how and when users leave a website. Much of website optimization entails trying to reduce drop-off rates.

Paths through the website: Finally, analysts are often interested in studying specific paths through a website. For example, an analyst might want to see all the paths that take visitors from the home page to the checkout page and how those are different for different types of users. They are interested not only in which pages were visited but also the sequence of those pages. For example, a path that contains many visits to the same webpage can be a sign of a design flaw or bug in the implementation.

There are a number of other important tasks that we do not address in this work, such as optimizing search engine performance or detecting looping behavior where visitors go back and forth between the same set of webpages.

DESIGN OF MATRIXWAVE

We now describe our visualization techniques for studying and comparing event sequence datasets. We discuss the design of the MatrixWave representation, and present techniques for embedding comparisons into the representation.

Visualizing a single dataset

The inputs are sets of event sequences, where each event sequence is a list of *nodes*. For example, we might have four possible nodes A, B, C, and D, and an individual sequence might be $B \rightarrow A \rightarrow C$. This path contains three *steps*. A transition between nodes at a given step is called a *link*. Nodes and links have traffic volumes associated with them. In clickstream visualization, each node represents a webpage view, and a sequence represents a single user's navigation history, e.g., start at page B, go to page A, go to page C, exit. A simple example dataset is shown in Figure 2-a.

Modern visualizations of clickstream data typically employ the Sankey diagram, which offers an aggregate view of paths and offers more details through interaction. A Sankey diagram aggregates clickstream data according to steps (Figure 2-b). At each step, identical pages are aggregated into nodes, and nodes are sized according to their volume. Links connect nodes between steps and are also sized according to volume. For clickstream data, drop-off links show the volume of traffic leaving the site at various steps.

While Sankey diagrams are popular, their major weakness is in handling dense transitions. Even with less than ten nodes and dozens of paths, a Sankey diagram is hard to read due to the multitude of edge crossings.

The approach we propose, MatrixWave, replaces the edge connections between a pair of steps with a transition matrix (Figure 2-c). Each cell in the matrix represents the volume of traffic between the corresponding two nodes. This is inspired by the observation that matrices can effectively display dense graphs [8]. We hypothesize that the advantages of the matrix representation still apply in the context of visualizing clickstream data. Applying the matrix representation to each step in the Sankey representation, we obtain a sequence of matrices placed next to each other in a sequence (Figure 2-d).

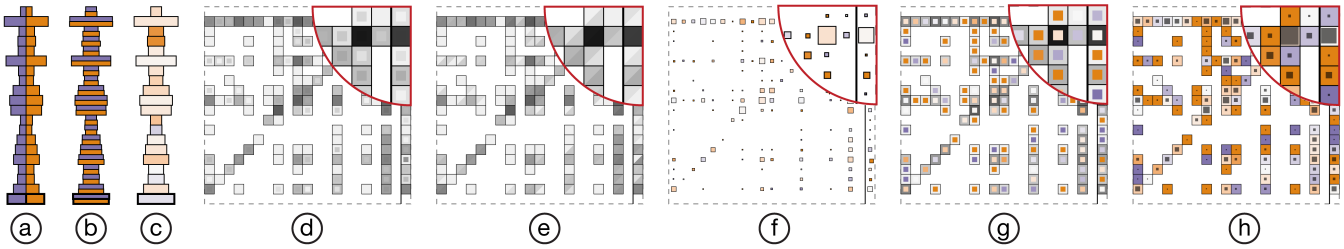

Figure 3. Examples of visual encodings developed in MatrixWave to facilitate data comparison of two event sequence datasets.

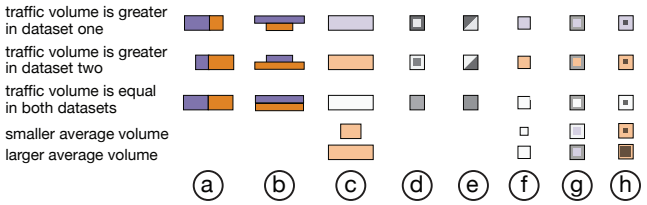

Figure 4. Potential designs for showing differences.

To represent drop-offs, we add an additional drop-off node to each step of the matrices.

One problem with this design is that the notion of a continuous path is lost because the nodes at each step are duplicated. To avoid duplication of nodes, we rotate the matrices 45 degrees and concatenate them. The rotation allows us to use nodes in each step to show both incoming and outgoing links (Figure 2-e). A user can then trace a path in a zig-zag manner.

Visual encodings for comparison

To do comparative analysis of two datasets, we must first place them into correspondence. We assume that most nodes are in common; e.g., the only changes to a website structure might be the addition or removal of a few pages. We match nodes that have the same page name and step number, and links that have the same source and target nodes. If a node or a link only appears in one dataset, the corresponding volume in the other dataset is set to zero. In real-world datasets, a step can include hundreds of distinct page events. We aggregate page events into groups using the website's organizing hierarchy.

We consider four approaches for visual comparison: juxtaposition, superposition, explicit encoding, and animation [9]. In the design of MatrixWave, we apply *superposition* to represent two datasets within one visualization. Showing two separate visualizations side-by-side (juxtaposition) is not effective in comparing complicated structures, because corresponding elements are far apart and difficult to identify [1]. While animations can provide smooth transitions between multiple datasets, they require switching between views repeatedly. Within a visualization, we apply *explicit encoding* to show volume differences in nodes and links. We explain the design of explicit visual encodings below.

Node encoding choices

We apply *explicit encoding* in MatrixWave, where the difference in traffic at each node is computed and directly displayed. It is common practice to map differences to a diverging color scheme, e.g., purple to orange (Figure 3-c), so that it is easy to identify the direction of change. Both relative difference and percentage difference can be mapped to the color-scale. In addition, we use size to show the average volume, as the importance of a difference is often determined by the volume of traffic. Changes on popular pages are more important than changes on pages with less traffic.

A design alternative is to use juxtaposition: for every page group, we have two nodes, each representing the page group in one of datasets. Size is an effective visual variable to encode volume and hue is appropriate for representing dataset membership. We can either align the two nodes along a center line (Figure 3-a) or put them side by side (Figure 3-b) for easier comparison. Figure 4-a to -c show how the potential designs represent differences. Juxtaposition requires analysts to make perceptual judgments about the differences in volume, while explicit encoding is more efficient and provides direct access to this information [9].

Link encoding choices

Differences in link traffic can also be displayed by juxtaposition or explicit encodings. Our final design uses *explicit encodings* with two concentric squares: the inner square size represents the average link volume of the two datasets, and the background hue of the cell encodes the difference in traffic volume (Figure 3-h).

We started by designing two types of glyphs for juxtaposition: one using two fixed-sized concentric squares (Figure 3-d), and the other dividing a cell along the diagonal into two triangles (Figure 3-e). Color intensity of the two regions represents link volume in the two datasets. Previous work has explored other glyphs for encoding differences and found that the square division method outperformed the rest [1].

We also tried to use color to explicitly encode difference and size to represent link volume (Figure 3-f), similar to the node encoding. Due to the limited number of pixels allocated to each cell, this approach results in tiny squares when the link volume is small, making it difficult to read the differences represented as color. We considered swapping the mapping (so that color represents volume and size encodes difference), but size is not effective at indicating the direction of difference, positive vs. negative.

To overcome this problem, we tried representing link volume with two concentric squares with two different colors (Figure 3-g), but found it harder to interpret two different colors representing different types of quantities. In the end, we decided to size a black square inside of each cell according to link volume, and use the hue of the background of each cell to represents the magnitude of difference (Figure 3-h). This design is also consistent with our choice of node encoding.

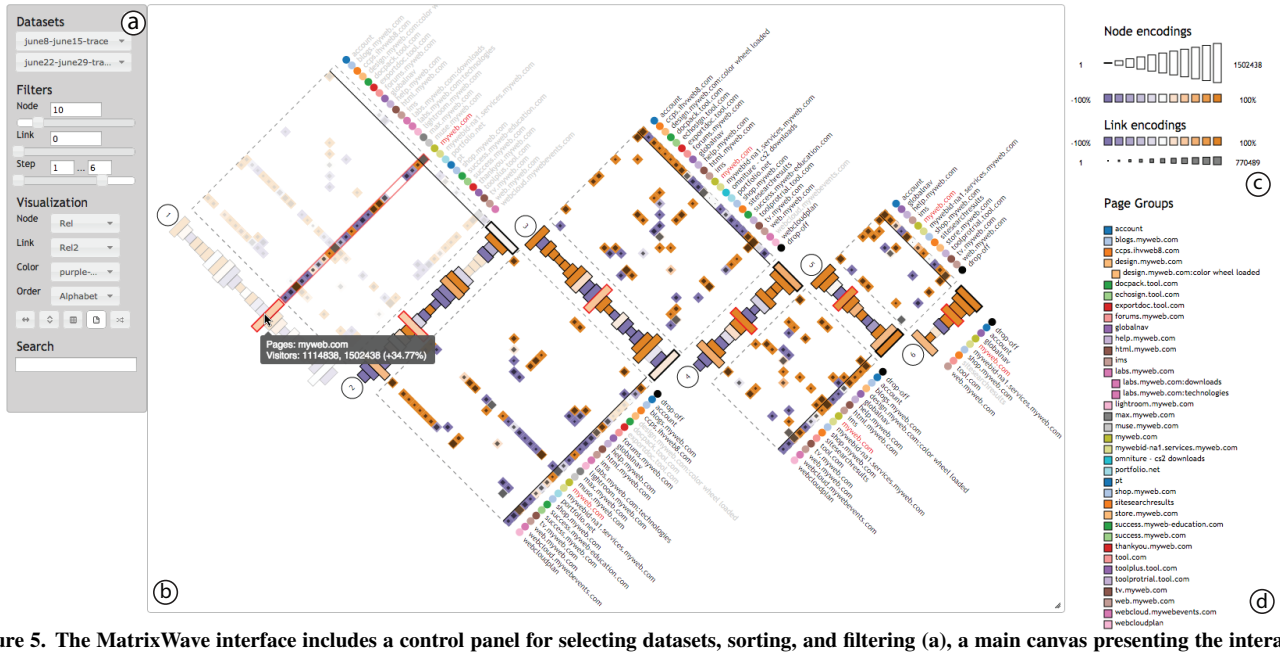

Figure 5. The MatrixWave interface includes a control panel for selecting datasets, sorting, and filtering (a), a main canvas presenting the interactive visualization (b), a legend (c), and a list of page groups (d). Here the user has highlighted a webpage from the first step. The bright orange and purple squares representing links show that there are lots of differences between the two datasets. The user can use the groups in the panel on the right (d) to filter the dataset according to the website structure.

Figure 4-d to -h show those design alternatives for comparing values in two datasets.

In our final implementation, MatrixWave visualizes the event sequence data with the designs shown in Figure 3-c and -h by default, using percentage difference as the measurement. Other alternatives can be accessed in the prototype through the control panel (Figure 5-a). In addition, we display the page group names associated with each group of nodes on the opposite side of each matrix (Figure 5-b).

Interactive Features

MatrixWave incorporates a number of interactions to further facilitate the exploration and comparison of event sequences.

Brushing & linking and selecting: When a user hovers over a node, he sees more information about that page group, including the traffic volume in both datasets and the difference as a percentage. Additionally all the incoming source nodes, outgoing target nodes, and the links across all steps are highlighted (Figure 5-b). Similarly, hovering over a link offers more details about the transition and highlights its source and target nodes. A user can click on a node or a link to select it. When a node or a link is selected, traffic through that node or link is highlighted. To allow users to move around the visualization, MatrixWave supports zooming and panning.

Showing paths: Since analysts are often interested in understanding user paths through a website, we provide a way to overlay path information on top of the matrix representation. When one or more nodes/links are selected, as shown in Figure 6-a, blue lines indicate the underlying event sequences that pass through the selected nodes/links.

The traffic volume of the selected paths is typically only a subset of the overall volume. For example, in Figure 2, if we select the path $A \rightarrow A \rightarrow B$, the third-step volume for B is only a fraction of the overall volume for B at that step. To represent these path-specific volumes, we overlay visuals with the same encodings. In particular, for a node, an inner bar representing the path-specific volume is displayed on top of the overall aggregate volume bar (Figure 6-b); and for a link, the matrix cell is split into two triangles where the right half denotes the path-specific volume and difference (Figure 6-c). This path-specific representation can be turned on or off through the control panel.

Filtering: We developed several filtering mechanisms in MatrixWave. First, users can search for page group names in a search box and select all the matching elements at once (Figure 6-a). This is useful when a user decides to focus on exploring the flow of a specific page group or a subset of page groups in the same name domain. Second, users can filter both nodes and links by volume. Additionally, entire steps can be filtered out to support more advanced exploration, for example, removing unnecessary intermediate steps when only the starting and ending nodes are of interest. Finally, pages can be filtered according to their organization in the website using the tree-structured page listing (Figure 6-d).

Ordering: In order to facilitate comparison, a user can sort page groups alphabetically, by volume, or difference. Sorting can be performed within a specific step or across all steps. Additionally, MatrixWave allows the user to insert proxy nodes for all page groups in the website at each step. This facilitates using spatial positions as anchors into the identity of a page (i.e. the third node in a step will always be the same page or page group). Using proxy nodes can result in very large and sparse matrices, but it makes it easier to get an overview of the entire site and to trace a particular page group across steps. Finally, a user can also fix the ordering

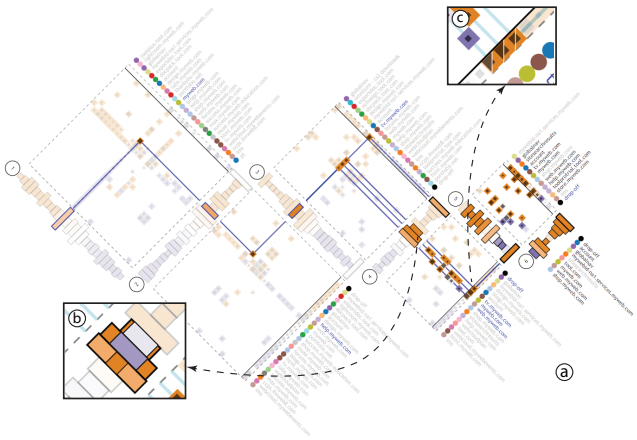

Figure 6. When one or more objects are selected, paths passing through the selected nodes/links are indicated with a blue-line (a). Path-specific differences are overlaid to show how the path volume is different from the aggregate volume (b)(c). In this example, the differences in the path specific volume of the highlighted nodes are the same as the differences in the aggregate nodes (b). The path-specific link information (c), shows that there is an increase in the number of path-specific links from one dataset to the other.

of the nodes in one step and propagate this ordering across steps. This facilitates comparison across steps.

ANALYST EXPERIENCE

We designed MatrixWave iteratively by working with web data analysts. To explain how analysts use MatrixWave let's walk through a typical workflow. For a complete view of the exploration process, please refer to the supplemental video.

John is examining a website following a major redesign. He loads two datasets into MatrixWave for comparison—one from the week before the redesign and one from the week after. At a glance, he sees a lot of differences: there are a lot of pages with bright orange and bright purple colors. He sorts the pages according to changes in traffic.

The homepage immediately pops out. It has many links (Figure 5), but many of them are purple, which means that traffic is down. He hovers over those links and sees that many of those links are no longer visited from the homepage. They are probably no longer on the homepage, John thinks. There is a 35% drop in traffic to the forums pages and a huge jump in traffic to the help pages (390%). He clicks on the link connecting the homepage and the help pages to highlight all traffic that passes through the homepage and the help pages.

From the help pages, he sees people go on to four different types of pages next. There is a huge jump in visits to the video site, where some of the help videos are hosted. He selects the videos site in step 3 to see what happens next (Figure 6). He is surprised to see that everyone leaves after step 4. So, visitors who go to the help site after the homepage, are leaving within a couple of pages. He looks at all the drop off nodes and sees that the drop-off nodes for step 4, 5, and 6 have seen a big increase. Looks like people are leaving the site faster now after the redesign. John needs to investigate further.

ADAPTING SANKEY FOR VISUAL COMPARISON

Since our work is one of the first to investigate comparative analysis of event sequences, there are no existing baselines to compare with our approach. To help us understand the strengths and weaknesses of MatrixWave, we chose to adapt the conventional Sankey diagram. The Sankey diagram is widely used for visualizing large numbers of event sequences in many applications, such as Google Analytics and [30].

We adapted the Sankey for comparison using superposition and explicit encodings (Figure 1-b). Nodes are represented in the same way as in MatrixWave. Unlike the traditional Sankey visualization, we placed the node bars horizontally in each step to allow for easier comparison of page volume. For encoding differences in links, we use smooth-curved ribbons colored according to difference magnitude. The thickness of the ribbon is mapped to the average of the two volumes associated with that link. We used the same divergent color scheme as in the MatrixWave design. Additionally, we replicated all interactions available in the MatrixWave prototype. To highlight paths, we used blue outlines for all the related nodes and links. To encode path-specific volume on top of nodes, we used the same representation as we did in MatrixWave. To encode path-specific volume in links, we extended the overlay paradigm and overlaid smaller ribbons on top of the aggregate ribbons.

EVALUATION

We conducted a laboratory study to investigate the effectiveness of MatrixWave in supporting visual comparison tasks. In the study, we compared MatrixWave with the adapted Sankey visualization we described above. The main goal of the study was to understand the advantages and disadvantages of these two distinct approaches and get feedback from both novice and expert users on their preferences. Since previous work has indicated that a matrix representation is more effective than the node-link diagram for larger and denser data [1, 8], we focused on comparative analysis tasks with two datasets, instead of comparing MatrixWave and the node-link based Sankey for visualizing a single dataset.

Participants and apparatus

We recruited 12 volunteers, 8 males and 4 females, aged 22–53 (mean 42), from a large IT company. Participants were from various technical backgrounds, such as software engineering, product management, data analysis, and graphics design. Six of them had familiarity with website clickstream data, and four considered data analysis part of their job. All participants had normal vision (or corrected-to-normal vision) without color deficiency. The experiments were conducted on a laptop (a MacBook Pro with 2.3 GHz Intel Core i7 CPU and 16 GB memory) connected to a 30-inch Dell desktop monitor. The visualization on the screen was restricted to an area of 1600×1000 pixels. All participants completed the study with a mouse and keyboard.

Data

We created our study datasets from real clickstream data captured on a large organization's website. We visualized the first 6 steps of the 1000 most common event sequences

on two different days, where the input data covered 90% of page groups in the entire traces. We used the first 6 steps, in order to allow the visualization to fit on the screen; the first few steps are usually the most important in web clickstream analysis. The final input to the visualization contained 167 nodes and 902 links in total.

Methodology

We performed a within-subject, full factorial design with two experimental conditions and eight tasks. Two different sets of eight task questions were developed to address any memory learning effects. The orders of the techniques and the task sets were counter-balanced using a Latin square. The order of the tasks within each set was fixed.

We created eight study tasks that cover a variety of interactions we saw analysts do with the visualizations during our interviews including: node comparison, link comparison, comparison across steps, and path comparison. Each task included a multiple-choice question and a visualization customized to the question. Customizations included sorting the nodes and highlighting specific nodes and links. As others have shown previously [1], eliminating the task of selecting related visual elements allows us to better isolate the effect visual encodings have on performance. Each task included a "do not know" option to prevent users from randomly guessing. The template we used for task questions is shown in Table 1. We carefully chose the data parameters in task questions to avoid ambiguous answers and to maintain the same level of difficulty across task sets. Each question had only one correct answer.

Node comparison tasks: We found node comparison within a step and across steps to be part of almost every analyst task. For example when looking at overall traffic volume, analysts focus mostly on nodes and look for big changes in color, especially for the popular pages that receive more traffic.

Link comparison tasks: Link comparison invokes comparing the number of outgoing and incoming links and how they are different across datasets. Link comparison is an integral part of understanding entry and exit of the website. To understand where people go when they get to the website, analysts look at the number of links and their connections.

Comparison across steps: Comparison across steps involves comparing nodes across steps and comparing links across steps. This type of comparison is part of understanding whether users are leaving the site faster in one dataset than the other and whether visitors in one dataset are visiting more of the website as compared to visitors from the other dataset.

Path comparison tasks: Path comparison involves selecting two or more nodes and looking at the specific paths between those nodes. In particular, analysts want to know if there are multiple paths and which nodes they include. Path comparison is rarely done today, because analysts do not have the right tools to do it.

Procedure

The study began with a brief tutorial to the participants about the problem domain and the visualizations. Participants then performed four training tasks with each of the techniques (Sankey and MatrixWave) with a separate dataset. During the training session, participants were instructed to think aloud, and the experimenter helped them resolve any questions. Next, the participants went through 8 tasks with each technique for a total of 16 tasks. Each tasks was limited to 2 minutes, after which the system automatically advanced to the next trial. We reminded participants that they could always select the "do not know" option when their confidence was low. We recorded task completion times and users' answers. We also observed how participants performed the tasks and captured the entire session using screen recording software. After the study, participants were asked to specify their preference for a visualization technique, and we conducted a semi-structured interview to collect their feedback. The whole study lasted approximately 1 hour for each participant.

	Type	Question
T1	Node	Pages A and B have been highlighted in steps x, y, and z. Which page has a larger overall volume?
T2	Node	Pages A and B have been highlighted in steps x, y, and z. Which page has a greater overall change?
T3	Link	Pages A and B are highlighted in step x. Which of these two pages has more incoming links with increasing volume?
T4	Link	Pages A and B are highlighted in step x. Among all the outgoing links from these two pages, which has the largest volume?
T5	Step	Is the overall traffic between the highlighted pages in steps x and $x+1$ increasing or decreasing?
T6	Step	Compare the traffic between highlighted pages at step x to $x+1$ and at step y to $y+1$. Which has a larger overall volume?
T7	Path	Observe the paths of traffic going through the highlighted pages. Which page has the largest path-specific volume at step x?
T8	Path	Observe the paths of traffic going through the highlighted pages. Is the overall path-specific volume increasing or decreasing?

Table 1. Experimental tasks.

Hypotheses

We had the following hypotheses about the performance of these two visualization techniques:

H1 For node comparison tasks, we expect Sankey to perform faster, because MatrixWave introduces a rotation making it more difficult to compare nodes; we expect the two techniques to achieve a similar accuracy, because they use the same visual encodings.

H2 For link comparison and step comparison tasks, we expect MatrixWave to perform faster and more accurately, because links are occluded by edge crossings in Sankey.

H3 For path tasks, we expect Sankey to perform faster than MatrixWave, because Sankey is more familiar to users and users do not have to follow a zig-zag to understand the path; for the same reason in H1, we expect the two techniques to achieve a similar accuracy.

Results

Accuracy

On average, MatrixWave achieved an accuracy of 95% (SD=6%) for all tasks, which was much higher than Sankey, which had an accuracy of 79% (SD=17%). Both incorrect answers and "do not know" choices were counted as

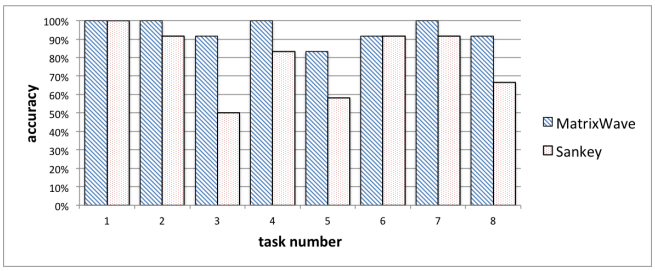

Figure 7. Accuracy of each task

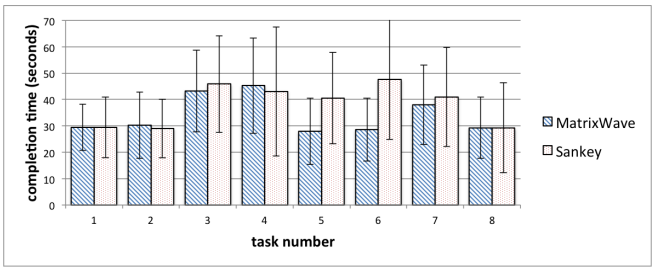

Figure 8. Completion time of each task (errors bars indicate the standard deviation).

incorrect. There was a significant effect of visualization techniques on task accuracy ($F_{1,14}$=5.918, p<0.01). Figure 7 shows a more detailed picture with accuracy for each task. Despite our expectations that Sankey would outperform MatrixWave for node and path tasks (H1 and H3), both techniques performed equally well for node tasks (T1, T2) and MatrixWave outperformed Sankey in path tasks (T7, T8). The greatest difference in accuracy was for link comparison tasks (T3, T4), where MatrixWave was about 30% more accurate, supporting H2. For the step comparison tasks (T5, T6), we found a different story. While MatrixWave outperformed Sankey for T5, it did about the same for T6. This result might be due to the nature of the task, which does not ask for precise comparison. The participants only had to determine whether one set of links is larger in volume than another set of links. In contrast, T4, which was also about link volume comparison, required users to specify which link was bigger. Thus, tasks that require more precision in link and step comparison are better supported by MatrixWave.

Completion time

We analyzed task completion time for all tasks that were completed correctly. On average, the participants spent 34.0 seconds (SD=16.2) on a task using MatrixWave and 38.2 seconds (SD=19.6) using Sankey. Figure 8 shows the completion time for each task. We found a significant effect ($F_{1,22}$=6.097, p<0.01) of visualization technique for the step comparison tasks (T5 and T6), which asked participants to assess the overall traffic difference between steps. MatrixWave was about 36% faster than Sankey for step comparison tasks, supporting H2. There was no significant effect for other task types. Sankey had an overall higher standard deviation on all tasks, especially in link and step tasks where we expected MatrixWave to be faster, indicating that participants may resort to random guessing more when using Sankey.

Preference and feedback

The preference ratings from all participants were: 2 strongly preferred MatrixWave, 6 preferred MatrixWave, 2 were neutral, and 2 preferred Sankey, indicating that 8 out of 12 participants liked MatrixWave. From the study interviews, we gathered many qualitative comments from participants about the advantages and disadvantages of both techniques.

First impressions: When we presented the two techniques to participants, people felt that the Sankey representation was easy to understand, because the nodes and the links of each step followed a linear left-to-right layout, which is commonly seen in representing data flows. However, participants also felt that it was overwhelming and noisy, because many links with different thicknesses, colors, and curvatures were overlapping with each other. As for MatrixWave, three participants found it difficult to comprehend the zig-zag layout of matrices at first, mentioning "*I need to think about it a bit and try to get used to it*". Some said that they had never seen the matrix representation of graphs and thought that using squares to encode links was less intuitive. However, all participants said that MatrixWave was more concise and clean, making information visible in a compact manner. They also liked the fact that MatrixWave provided a clearer overview of the data, saying that "*the matrices are like a heatmap, clearly telling me what is happening in which part of the website*".

Learning: Although many participants were not familiar with the matrix representation of a graph and the layout of matrices in MatrixWave, they could interpret the visual representations after some explanation. For both visualizations, we also observed that the visual encoding of volume and differences were easy for participants to understand. They did not have any questions about the volume and difference visual encoding following the training tasks. The path-specific visual encoding was more difficult for users, because it required differentiating between aggregate flow and path-specific flow. One user suggested that "*showing only the paths and removing the overall ones could be better*". Four participants had a dramatic change in preferences over the course of the study. During the training sessions, they said to themselves (the think aloud protocol) that Sankey was a lot better than MatrixWave, but after the study, they all agreed that MatrixWave was more useful and preferable.

Ease of use: As expected (H2), participants mentioned that it was extremely difficult to perform link-related tasks with Sankey, because 1) there were too many edge crossings, making the identification of specific links tedious and sometimes impossible, and 2) it was hard to compare the thickness of links with different curvatures. Participants thought the visual design in MatrixWave addressed these problems, commenting that "*it is clearer to see the incoming and outgoing links in the matrix because they are all aligned to the same row [or column]*". Moreover, for path-related tasks, participants generally thought that tracing multiple paths in MatrixWave was much easier with the blue line as a visual cue, even though the Sankey diagram applied similar highlighting. Two participants mentioned that MatrixWave design was evocative of architectural design with the matrices as rooms. One participant, who is an experience designer, said that she felt this design would help her explain the visitor's journey

through the website in a way that the Sankey diagram doesn't. Another participant said: "*it [a path] looks like a person enters a room [the matrix] through a door [the node] and find a table [the link square] and exits it through another door*". Participants also commented that the page labels were difficult to see in the Sankey diagram because they overlapped with the link ribbons, whereas MatrixWave displayed them next to the matrices. But in MatrixWave people needed to rotate their heads to read the labels, which some mentioned was uncomfortable.

Domain-specific usage: From the participants who had experience with analyzing web clickstream data, we collected a number of domain-specific comments. For example, one mentioned that the visual design of encoding volume and differences was effective for real-world cases, because people could related the two aspects to pin-point which pages to examine further, e.g., moderate differences on large-volume pages are more important than large differences on small-volume pages. Another comment was that when pages in each step were properly ordered, the matrix representation in MatrixWave showed a heatmap of traffic across different sections of a website, so that "*hot regions*" would pop up immediately. Moreover, participants pointed out that it was extremely useful to be able to select nodes in examining traffic for anticipated user behaviors. For example, an analyst could select certain entry pages and drop-off points to explore the volumes and differences of the connecting paths. Further, when paths were highlighted, participants thought that the spatial locations of nodes and link cells were strong visual cues indicating where the transitions happened in the website.

DISCUSSION

The results of our laboratory study indicate that MatrixWave is more effective than Sankey for visual comparison tasks. Except for node tasks where we expected the Sankey to outperform MatrixWave, MatrixWave was either more accurate or significantly faster than Sankey.

For link tasks, MatrixWave and Sankey achieved similar completion times, which was a bit surprising, because we expected that edge crossings would lower the performance of Sankey in both time and accuracy. It could be that highlighting/fading out links and nodes helped to resolve the visual clutter common in Sankey. However, even with highlighting, Sankey's accuracy rate is much lower than that of MatrixWave. This effect became more pronounced when participants had to assess a large number of links. For example, in step tasks, MatrixWave significantly outperformed Sankey in both speed and accuracy rate. The results for path tasks were also surprising. Despite the zig-zag paths, MatrixWave achieved a similar completion time and was more accurate than Sankey. Additionally, users preferred the visual representation of paths in MatrixWave in the form of blue lines. Further research is needed to explore different highlighting techniques and how those interact with the visualization for path tasks.

In practice, one should consider many factors beyond just completion time and errors in making design decisions. For example, if the data is sparse and small, a Sankey diagram might be preferable, because it is familiar to users and will not have much visual clutter. Also, if node or step comparison tasks are most important, the Sankey diagram might be sufficient. But, if the dataset is large and dense, MatrixWave is significantly more effective for all levels of comparison tasks once users become familiar with the visual encodings.

One concern with the MatrixWave design is the layout and the rotated matrices. Although the current design offers the best screen real-estate utilization, it also introduces usability issues. For example, users have to compare nodes in a step along a diagonal axis. The experimental results for node comparison tasks seem to indicate that this is not a big factor for performance, however, participants reported discomfort reading the rotated page labels. In future work we plan to explore how we might use interaction to ameliorate some of this discomfort. For example, we can allow users to rotate an individual matrix in place or the whole visualization when necessary. Another solution is to draw the text labels horizontally. However some part of the text may be obscured by the visualization.

In MatrixWave, we did not encode temporal information beyond order. Although in the exploration of web clickstream data, analysts mainly focus on the logical time of events, i.e., steps, temporal information, such as the duration of a event, is often useful in analyzing event sequence data in other application domains. To represent event durations, MatrixWave can easily be extended to include visual encodings similar to the "time edges" in OutFlow [30] next to a node. In addition, to represent event timestamps, a short line can be drawn on top a node with its position mapped to the delay of the event relative to a reference timeframe.

CONCLUSION AND FUTURE WORK

We have presented MatrixWave, a visualization technique for comparing two event sequence datasets. MatrixWave displays step-by-step clickstream data as a series of transition matrices rotated and concatenated in a zig-zag manner. By involving expert web clickstream analysts in the design process, we explored a number of design alternatives to visually represent two datasets using superposition and explicit encoding. Despite a steep learning curve, participants in a lab study preferred MatrixWave over the conventional Sankey and were more accurate in completing comparison tasks.

There are a number of promising future directions. First, we plan to further evaluate MatrixWave and study how domain experts use it in their everyday work. Second, we want to pursue comparison across multiple datasets. For example, it would be great to be able to show how traffic patterns have changed over an entire week rather than just over two days. Finally, we want to combine multiple event sequence visualization techniques including the Sankey, MatrixWave, and Icicle plot [31] into one system and study how they can be used to together to solve tasks and how analysts can smoothly move from one representation to another.

REFERENCES

1. B. Alper, B. Bach, N. Henry Riche, T. Isenberg, and J.-D. Fekete. Weighted graph comparison techniques for brain connectivity analysis. In *Proc. CHI*, 483–492, 2013.
2. B. Bach, E. Pietriga, and J.-D. Fekete. Graphdiaries: Animated transitions and temporal navigation for dynamic networks. *IEEE TVCG*, 20(5):740–754, 2014.
3. J. Bertin. *Semiology of Graphics: Diagrams, Networks, Maps*. University of Wisconsin Press, 1983.
4. A. Bezerianos, P. Dragicevic, J.-D. Fekete, J. Bae, and B. Watson. GeneaQuilts: A System for Exploring Large Genealogies. *IEEE TVCG*, 16(6):1073–1081, 2010.
5. J. Brainerd and B. Becker. Case study: e-commerce clickstream visualization. In *Proc. InfoVis*, 153, 2001.
6. S. Bremm, T. von Landesberger, M. Hess, T. Schreck, P. Weil, and K. Hamacherk. Interactive visual comparison of multiple trees. In *Proc. VAST*, 31–40, 2011.
7. I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Visualization of navigation patterns on a web site using model-based clustering. In *Proc. KDD*, 280–284, 2000.
8. M. Ghoniem, J.-D. Fekete, and P. Castagliola. On the readability of graphs using node-link and matrix-based representations: A controlled experiment and statistical analysis. *Info. Vis.*, 4(2):114–135, 2005.
9. M. Gleicher, D. Albers, R. Walker, I. Jusufi, C. D. Hansen, and J. C. Roberts. Visual comparison for information visualization. *Info. Vis.*, 10(4):289–309, 2011.
10. D. Gotz and H. Stavropoulos. Decisionflow: Visual analytics for high-dimensional temporal event sequence data. *IEEE TVCG*, 20(12):1783–1792, 2014.
11. J. Guerra-Gomez, M. Pack, C. Plaisant, and B. Shneiderman. Visualizing change over time using dynamic hierarchies: Treeversity2 and the stemview. *IEEE TVCG*, 19(12):2566–2575, 2013.
12. D. Holten and J. J. van Wijk. Visual comparison of hierarchically organized data. *Comp. Graph. Forum*, 27(3):759–766, 2008.
13. J. I. Hong, J. Heer, S. Waterson, and J. Landay. Webquilt: A proxy-based approach to remote web usability testing. *IEEE Trans. Info. Sys*, 19(3):263–285, 2001.
14. S. Kannappady, S. P. Mudur, and N. Shiri. Visualization of web usage patterns. In *Proc. IDEAS*, 220–227, 2006.
15. M. Krstajic, E. Bertini, and D. Keim. Cloudlines: Compact display of event episodes in multiple time-series. *IEEE TVCG*, 17(12):2432–2439, 2011.
16. B. Lee, G. G. Robertson, M. Czerwinski, and C. S. Parr. Candidtree: visualizing structural uncertainty in similar hierarchies. *Info. Vis.*, 6(3):233–246, 2007.
17. J. Lee, M. Podlaseck, E. Schonberg, and R. Hoch. Visualization and analysis of clickstream data of online stores for understanding web merchandising. *Data Mining and Knowledge Discovery*, 5:59–84, 2001.
18. E. Maguire, P. Rocca-Serra, S.-A. Sansone, J. Davies, and M. Chen. Visual compression of workflow visualizations with automated detection of macro motifs. *IEEE TVCG*, 19(12):2576–2585, 2013.
19. M. Monroe, R. Lan, H. Lee, C. Plaisant, and B. Shneiderman. Temporal event sequence simplification. *IEEE TVCG*, 19(12):2227–2236, 2013.
20. M. Monroe, R. Lan, J. Morales del Olmo, B. Shneiderman, C. Plaisant, and J. Millstein. The challenges of specifying intervals and absences in temporal queries: A graphical language approach. In *Proc. CHI*, 2349–2358, 2013.
21. T. Munzner, F. Guimbretière, S. Tasiran, L. Zhang, and Y. Zhou. Treejuxtaposer: scalable tree comparison using focus+context with guaranteed visibility. *ACM Trans. Graphics*, 22(3):453–462, 2003.
22. A. Perer and J. Sun. Matrixflow: Temporal network visual analytics to track symptom evolution during disease progression. In *Proc. AMIA*, 716–725, 2012.
23. A. Perer and F. Wang. Frequence: Interactive mining and visualization of temporal frequent event sequences. In *Proc. AVI*, 153–162, 2014.
24. C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Shneiderman. Lifelines: visualizing personal histories. In *Proc. CHI*, 221–227, 1996.
25. P. Riehmann, M. Hanfler, and B. Froehlich. Interactive sankey diagrams. In *Proc. InfoVis*, 233–240, 2005.
26. Y. Tu and H.-W. Shen. Visualizing changes of hierarchical data using treemaps. *IEEE TVCG*, 13(6):1286–1293, 2007.
27. F. van Ham, H. Van De Wetering, and J. van Wijk. Interactive visualization of state transition systems. *IEEE TVCG*, 8(4):319–329, 2002.
28. T. D. Wang, C. Plaisant, A. J. Quinn, R. Stanchak, S. Murphy, and B. Shneiderman. Aligning temporal data by sentinel events: Discovering patterns in electronic health records. In *Proc. CHI*, 457–466, 2008.
29. J. Wei, Z. Shen, N. Sundaresan, and K.-L. Ma. Visual cluster exploration of web clickstream data. In *Proc. VAST*, 3–12, 2012.
30. K. Wongsuphasawat and D. Gotz. Exploring flow, factors, and outcomes of temporal event sequences with the outflow visualization. *IEEE TVCG*, 18(12):2659–2668, 2012.
31. 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 *Proc. CHI*, 1747–1756, 2011.
32. J. Zhao, F. Chevalier, C. Collins, and R. Balakrishnan. Facilitating discourse analysis with interactive visualization. *IEEE TVCG*, 18(12):2639–2648, 2012.
33. J. Zhao, S. M. Drucker, D. Fisher, and D. Brinkman. Timeslice: Interactive faceted browsing of timeline data. In *Proc. AVI*, 433–436, 2012.