Patterns and Sequences: Interactive Exploration of Clickstreams to Understand Common Visitor Paths

Zhicheng Liu, Yang Wang, Mira Dontcheva, Matthew Hoffman, Seth Walker and Alan Wilson

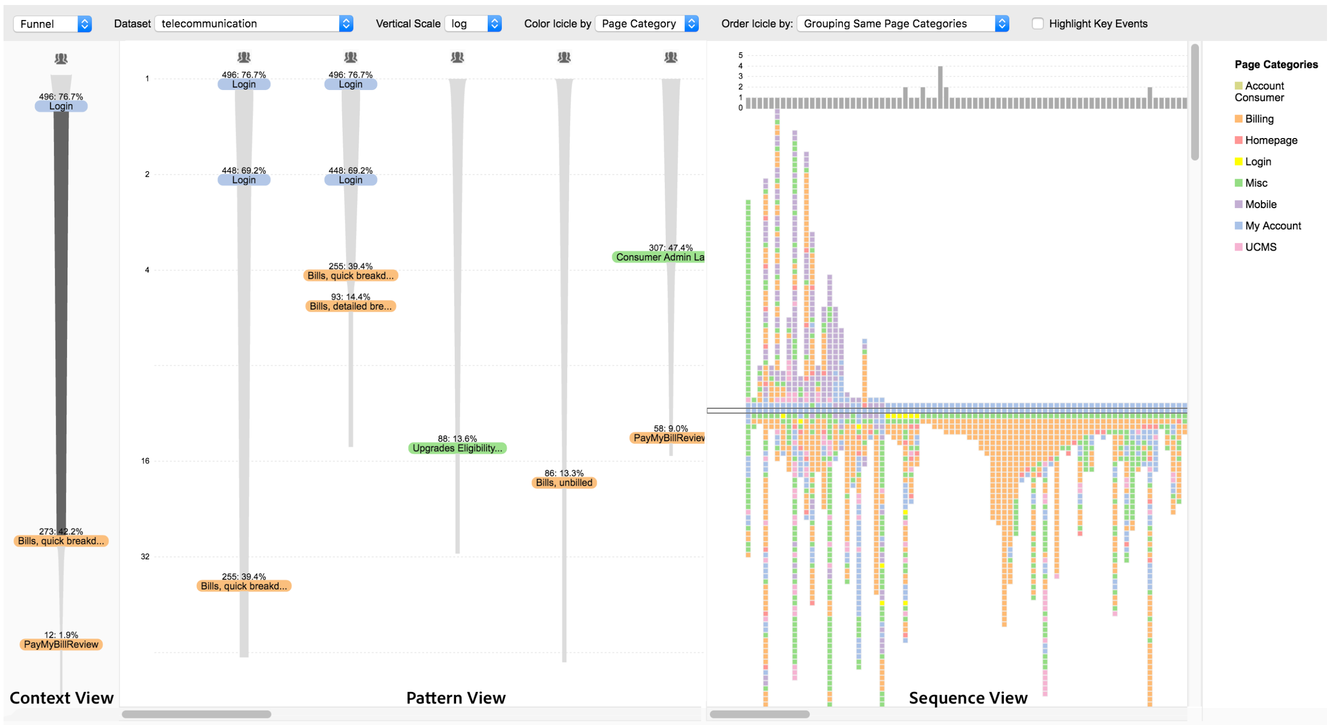

Funnel
Dataset telecommunication
Vertical Scale log
Color Icicle by Page Category
Order Icicle by: Grouping Same Page Categories
Highlight Key Events
496: 76.7% Login
273: 42.2% Bills, quick breakd...
12: 1.9% PayMyBillReview
Context View
448: 69.2% Login
255: 39.4% Bills, quick breakd...
93: 14.4% Bills, detailed bre...
88: 13.6% Upgrades Eligibility...
86: 13.3% Bills, unbilled
307: 47.4% Consumer Admin La
58: 9.0% PayMyBillReview
Pattern View
Sequence View
Page Categories
Account Consumer
Billing
Homepage
Login
Misc
Mobile
My Account
UCMS

Fig. 1: Interface design for interactive clickstream analysis: the pattern view shows maximal sequential patterns extracted from the dataset; the sequence view displays raw sequences in coordination with user interaction in the pattern view; the context view provides contextual information on the segment and hierarchical level of the dataset being explored.

Abstract—Modern web clickstream data consists of long, high-dimensional sequences of multivariate events, making it difficult to analyze. Following the overarching principle that the visual interface should provide information about the dataset at multiple levels of granularity and allow users to easily navigate across these levels, we identify four levels of granularity in clickstream analysis: patterns, segments, sequences and events. We present an analytic pipeline consisting of three stages: pattern mining, pattern pruning and coordinated exploration between patterns and sequences. Based on this approach, we discuss properties of maximal sequential patterns, propose methods to reduce the number of patterns and describe design considerations for visualizing the extracted sequential patterns and the corresponding raw sequences. We demonstrate the viability of our approach through an analysis scenario and discuss the strengths and limitations of the methods based on user feedback.

Index Terms—Clickstream Data, sequence mining, visual analytics, event sequences

1 Introduction

Companies and individuals collect huge amounts of clickstream data from websites and applications, in the hope that this data will allow them to better understand users' behavior and intentions. These clickstreams consist of series of ordered events triggered by user interactions. For example, e-commerce websites track visitors' browsing history to infer their intent and preferences for more effective online marketing, and application builders log user behavior to uncover usability problems and guide users to new product features.

To analyze such data, researchers have proposed a variety of visualization techniques [15, 23, 38] and sequence mining methods [2, 8, 25]. Visual analytic approaches seek to integrate visual exploration with automatic computation for real-world analytic tasks [13, 16, 21, 26, 32].

Clickstream analysts need to understand common paths taken by users, but doing so remains a significant challenge. Due to the high cardinality of the events and the long sequences, naive visual exploration of raw sequences is often inadequate to extract high-level struc-

- *Zhicheng Liu, Mira Dontcheva, Matthew Hoffman and Seth Walker are with Adobe Research. E-mail: {leoli,mirad,mathoffm,sewalker}@adobe.com.*
- *Yang Wang is with University of California, Davis. E-mail: ywang@ucdavis.edu.*
- *Alan Wilson is with Adobe Systems Inc. E-mail: alawilso@adobe.com.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx/

tures. While the data mining literature has contributed algorithms to summarize sequential data, there is little guidance on how these algorithms may be used in actual interactive analysis. On the other hand, visual analytic researchers have largely focused on providing statistical summaries of user-defined patterns. Few have investigated the issue of combining data mining and visualization design in depth.

In this paper we propose methods for interactive visual analysis of clickstream data, with a focus on understanding common paths shared by visitors. Applying a user-centered design process, our work makes three main contributions:

1. Following the principle to present and explore data across multiple levels of granularity, we identify **four levels of granularity** (patterns, segments, sequences and events) in the context of analyzing clickstreams as a result of several design iterations.

2. We propose a **three-stage analytic pipeline** for clickstream analysis: pattern mining, pattern pruning and coordinated exploration between patterns and sequences. In the current implementation, we focus on mining maximal sequential patterns (MSP), discuss the behavior of the mining algorithm and present techniques to prune the output space for visual presentation. We then describe design considerations for visualizing the extracted sequential patterns and the corresponding raw sequences.

3. To support exploration across abstraction levels, we design **novel visualization and interaction techniques** in a dual view interface. Analysts can align and segment sequences based on key events in sequential patterns. In addition, we introduce hierarchical pattern mining to help analysts focus on the desired level of granularity.

We illustrate the utility of these methods in an analysis scenario and discuss their strengths and limitations based on user feedback.

2 Methods

This research is the result of a longitudinal study with three analysts and two product managers over a period of three years. The three analysts work for a software company that sells applications and services online. Their daily jobs involve analyzing and reporting visitor behaviors on the company's website. They write queries to retrieve clickstream data from Hadoop servers, wrangle and analyze the data inside Excel, and prepare presentations on any insights they have found. The two product managers work on analytics products for digital marketing. We included product managers because they work with a much larger set of users and offer a high-level perspective on the common tasks shared by analysts from different companies.

To understand the scale of real-world data and the current practice of clickstream analysis, we conducted in-depth interviews with each of the analysts. The interviews were semi-structured, where we asked the analysts to explain the characteristics of the datasets they were working on, and the tools they were using. To get a first-hand experience of what their workflows were like, we performed the same analysis routines ourselves on the same datasets using the same tools. We also asked them to enumerate a list of specific questions that they would like to answer, and to identify those that were not supported by current tools. These exercises helped us exclude feature requests that could be addressed with engineering efforts and focus more on research challenges specific to visual analytics.

After identifying an initial set of analytical goals and tasks, we embarked on an iterative process to explore design ideas, implement prototypes and refine task specifications. We realized that mockups were insufficient to convey how a design might be useful to the analysts, because mockups were often based on small and often simplified datasets, and did not translate well to account for the complexity in real-world datasets. Consequently, we adopted a strategy of building interactive prototypes to explore the design space and get feedback from the users.

3 Data, Tasks and Existing Tools

Based on our observations and the interviews, we find that clickstream data has three distinguishing characteristics:

The cardinality of the event set is large. The number of unique events for a modern website can range from thousands to tens of thousands. Most events correspond to a click that loads a web page, but sometimes a single click can also trigger multiple system events.

The events are multivariate. Figure 2 shows a typical clickstream recording the behavior of a single visitor. The complete event names reflect the hierarchical organization of a website. For example, an error page generated due to an invalid input email is grouped under three levels of hierarchy: Account $\rightarrow$ IMS $\rightarrow$ signup. In addition to these hierarchical page names, each event can be associated with a number of variables such as device type (mobile or desktop), browser or geographic location.

The sequences are long. It is common for a single session to generate hundreds of events. Companies also use cross-device trackers to join visitor sessions across multiple devices, yielding even longer sequences. The long sequence length and the high-cardinality event set imply that few clickstream sequences are identical.

```
1: mybusiness.com:solutions:web
2: design.mybusiness.com:Plans:PageView
3: mybusiness.com:solutions:web
4: design.mybusiness.com:Plans:PageView
5: design.mybusiness.com:Plans:web:Annual:SignUpButton:Click
6: design.mybusiness.com:web:Join:1:IDForm:Page
7: Account:IMS:onLoad_SignUpForm
8: Account:IMS:SignUp:Error_InputValidEmail
9: Account:IMS:onLoad_SignUpForm
10: Account:IMS:SignUp:Error_AccountAlreadyExists
11: Account:IMS:onLoad_SignUpForm
12: Account:IMS:SignUp:Error_AccountAlreadyExists
13: Account:IMS:onLoad_SignInForm
14: Account:IMS:onLoad_ReturningUserSignedInSuccessfully:Remember_Me_Checked
... ... ... ...
58: design.mybusiness.com:DownloadCenter:DownloadSurvey-SignedIn:ltr
59: design.mybusiness.com:DownloadCenter:DownloadSurvey-SignedIn:Continue:Click:ltr
60: design.mybusiness.com:Authenticated
61: design.mybusiness.com:DownloadCenter:Download:PageView:ltr
62: design.mybusiness.com:DownloadCenter:DownloadClick:InstallAAM:ltr
63: helpx.mybusiness.com:ltr:how-to:ltr-mobile
64: design.mybusiness.com:Authenticated
... ... ... ...
```

Fig. 2: A single clickstream with event name at each step, demonstrating the length of sequence and the high event cardinality.

Due to the high event cardinality and long sequences, simple data aggregation and visualization techniques do not work well in supporting exploratory analysis. Figure 3 shows a variant of the icicle plot [15], visualizing 250 clickstreams. The events are color-coded by event categories, and each clickstream starts from the top to the bottom. Visualizing raw clickstreams using such visualizations is not useful for the analysts: the limited screen real estate entails that we can not display all the sequences without scrolling, even if we use pixel-based techniques [12]; similarly, the long sequences are often cut off, and the view port can only show sequence segments. Even with event aggregation, the visualizations often contain too much noise. While it may be possible to spot outliers (e.g. the long green sequences on the right), it is difficult to identify high-level salient patterns.

Since there is no effective way to provide a visual overview of the data, the analysts often resort to visualizations showing summary

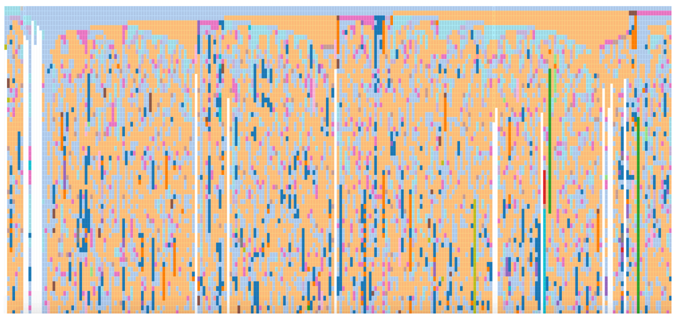


Fig. 3: A visualization of 250 clickstreams. Each rectangle represents an event, and the order of events goes from top to bottom. The color of the rectangles represents event category.

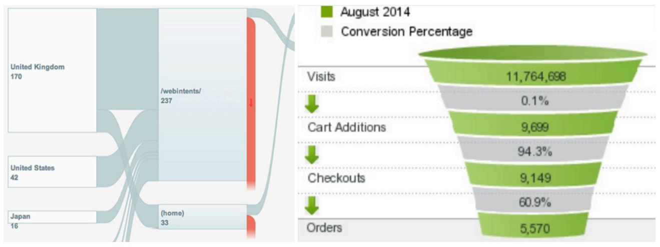


Fig. 4: (a) A flow visualization based on the Sankey diagram in Google Analytics, showing top pages at the first few steps in the clickstreams. (b) A funnel visualization from Adobe Analytics showing the drop off of visitors at key pages defined by analysts. The green areas represent the absolute number of visitors at each checkpoint, and the grey areas represent the percentage of visitors from the previous stage reaching the next stage.

statistics of the sequences. Two commonly used tools are the previous/next event flow visualization and the fall out visualization. Commercial applications such as Google Analytics and Adobe Analytics offer both types of visualizations. The previous/next event flow tool presents the top events before or after a few user-chosen events in the form of a Sankey diagram (Figure 4a). Such visualizations still suffer from the problem of only showing a small subset of the events and paths, and can result in visual cluttering [38]. Alternatively, the analysts can define a funnel, which consists of a sequence of events that the visitors are expected to go through. The fall out visualization shows summary statistics of the number of visitors going through the funnel (Figure 4b). It offers insights on the proportion of visitors reaching each key event in the funnel, but the analysts have no idea what is happening between each pair of events in the funnel, neither can they drill down to examine detailed information on individual sequences.

In short, these existing tools provide limited support for analyzing clickstream data, and the analysts describe the following tasks they need help with:

1. Identify key customer journeys: The analysts confirm that it is impossible to see all the sequences in their entirety. They are most interested in understanding the common paths taken by visitors and want to see an overview of visitors' navigation patterns. One analyst remarks: "*I want a visualization that tells me, this is probably what's happening most of the time: 87% of the time, people who clicked on page A also clicked on page B*".

2. Drill down into individual sequences: While showing every sequence will be overwhelming, the analysts still want to be able to examine individual events in sequences of interest. The analysts describe a tool that allows them to start at a high level and then drill down so that they can see sequences across multiple levels of granularity.

3. Bring dimensions into analysis: Since the events are multivariate, it is important to understand sequences of events from different perspectives, as reflected in event names or device information.

4. Reveal multiple occurrences of events: An event can repeat multiple times in a sequence. This kind of multiple occurrences often indicate loops in visitors' navigation patterns and may point to potential inefficiency or bugs in website design. Current tools often collapse such multiple occurrences into aggregated count and hide such information. The analysts want a visualization that reveals such patterns if they exist in the dataset.

4 Related Work

In principle, visualization and analytic methods for event sequence data are generally applicable to clickstream data. Previous work on visualizing event sequence data often aggregates raw sequences into a tree and applies hierarchical visualization techniques such as Sunburst [30] and the icicle plot [15]. Alternatively, we can aggregate events within each step and visualize the data using flow diagrams (e.g. the Sankey diagram [28] and the alluvial diagram [29]) and matrix-based approaches [38].

These techniques of visualizing raw sequences with simple data transformation rarely work on complex and high-volume real-world datasets. Instead of showing the analysts an overview of the data, query-driven approaches to event sequence analysis find matching sequences to user-defined patterns and show summary statistics as visualizations [14, 16, 37]. While such approaches are effective to answer specific questions, the analysts need to be able to anticipate potential patterns that exist in the dataset and may miss valuable insights.

Visual analytics approaches to event sequence analysis seek to combine the strengths of automatic computation with the flexibility of interactive exploration. Wei et. al. use a self-organizing map to cluster and visualize clickstream data [34]. Similarly, Wang et. al. perform unsupervised clustering on clickstream data by partitioning a similarity graph [33]. We considered using clustering to perform data reduction but encountered interpretation and trust issues in earlier iterations of our visualization design (Section 6). Unsupervised clustering has other downsides: it works best with a machine learning expert in the loop to fine-tune the parameters, and requires a domain expert to provide sensible cluster labels. Maguire et. al. extract motifs from scientific workflow data for better visual abstractions [20]. However, motif extraction is less effective on clickstream data given the high event cardinality and variation.

Mining frequent sequential patterns, on the other hand, is a promising data reduction approach that offers interpretable algorithmic parameters and results. FP-Viz is one of the earlier works that visualizes frequent itemsets mined from the data using the Sunburst technique [30]. CoCo [21] and the High-Volume Hypothesis Testing method [22] examine the problem of cohort comparison by extracting subsequences, but do not explore in depth how to design for non-consecutive sequences. Frequence [26] focuses on mining frequent patterns and provides a visual interface for pattern exploration. The main contribution of Frequence is the enhancements to the SPAM algorithm [2] to handle real-world constraints, and the interface makes use of conventional flow visualizations. In this paper, we choose the VMSP algorithm [7] instead of the SPAM algorithm to mine a more compact set of patterns, and we focus more on visualization and interaction design.

5 Data Preprocessing: Event Categorization

Given the high cardinality of events, our first step was to aggregate individual events into a manageable number of categories. We focused on event names since other dimensions of events often have a small set of predefined categories (e.g. device type) or a natural hierarchy (e.g. geo location). Our first attempt was to tokenize the event names and apply text-mining methods to extract meaningful categories. We evaluated the following three approaches:

Categorize by (sub)domain name: Since most event names are essential URLs with separators denoting a domain hierarchy (e.g. "mybusiness.com:solutions:web"), a naive approach is group the events by domains. However, we found that the design of a website's organization does not always reflect meaningful user intent or activities. For example, over 30% of the events in one of our datasets are under the root domain name (www.companyname.com). It is impossible to distinguish between these events under this naive approach.

Categorize by token frequency: Another approach is to consider tokenized event entries as a group of terms. We first adopted term frequency-inverse document frequency (tf-idf) to extract the representative terms, we then use those high-frequency terms as features to describe the events. This approach again failed to obtain meaningful categories, since the top-level domain names dominated the frequency terms, while the more informative event indicators were buried in deeper segments.

Categorize by topics: We also tried applied the Latent Dirichlet Allocation (LDA) model [3] to separate event entries into distinct categories. LDA associates each event with a "topic", and tries to choose these topics so that semantically related events are associated with the same topic. Our hope was that we could use these topics as higher-level event types. Unfortunately, 90% of the events fell into a single topic, making the results unusable.

Ultimately, we had to work with analysts to manually devise rules to categorize the events. Although none of the above unsupervised techniques yielded acceptable categorization results out-of-box, they served as good references that allowed us to accelerate the manual labeling process with the analysts.

6 Choosing Higher Levels of Granularity

The complexity and scale of datasets entail the need to first summarize or reduce the data before visualizing it. In this section, we discuss design rationales and lessons learned from our earlier exploration with various data mining techniques and the corresponding visualization designs. This exploration is guided by an overarching principle abstracted from Tasks 1 and 2 in Section 3: *the visual interface should provide information about the dataset at multiple levels of granularity , and allow users to easily navigate across these levels*. This principle has been articulated as a general guideline for visual analytics research [11] and deployed in prior work on system design [17, 24]. In the case of clickstream analysis, two levels of granularity are a given: events and sequences. The key challenge here is to pinpoint *higher granularity levels* that meet the following criteria: first, human users should be able to easily interpret the meaning of these granularity levels; second, they need to understand how these levels relate to each other; finally, information presented at these levels are potentially interesting or useful. We now review the data mining methods and evaluate them using these criteria.

6.1 Motif Extraction

Motifs (or *n*-grams) [20] are lists of consecutive events that frequently appear in event sequences. We first thought of extracting motifs to give analysts a high-level picture of the data. We extracted *n*-grams with *n* ranging from 2 to 5, and presented the motifs in a list. Users can sort the motifs either by length or by frequency. When we computed motifs on sequences aggregated by event categories, the extracted motifs are often dominated by transitions between events from the same category. Such patterns make sense since consecutive events are often triggered on the same web page of the same functionality, but are not useful.

We thus tried extracting motifs directly on raw sequences. The analysts considered the extracted motifs easy to understand. They also wanted to examine the motifs in the context of raw sequences, so that they could see what happened before and after the motifs. One problem with the motifs was that most of them were not really interesting. For example, variations of the checkout funnel (“load cart”, “payment info”, “payment confirmed”) and the login processes are typical motifs we extracted from the data, but they are hardly surprising.

6.2 Sequence Clustering

Another obvious way to summarize sequences is clustering. We tried three clustering methods: bag-of-events, bag-of-motifs and mixture of Markov Chains. The bag-of-events model ignores sequence information and only considers the relative frequencies of events. Although the resulting clusters did reveal event distributions (e.g. one cluster contained all sequences containing tutorial pages, while another featured many product information pages), the order in which the users visits the website was lost. The bag-of-motifs model extracts motifs and uses them as features for the clustering. The mixture of Markov Chains (MMC) model assumes pre-existing clusters defined in terms of a first-order Markov chain, and uses an expectation-maximization approach to compute the probabilities of a sequence being associated with each cluster. Each sequence is put into the cluster with the highest probability. One advantage of the MMC model is that it characterizes less-frequent transitions in the event sequences, which can be of potential interest to analysts.

To visualize the clusters, we show them as individual blocks (Figure 5). Due to the sheer size of the event sequence dataset and the limited screen space, only portions of selected sequences within each cluster can be shown. To reveal the most representative sequences within each cluster, we chose to sort the sequences by their affinity coefficients. We also provided multiple interactions to help reduce

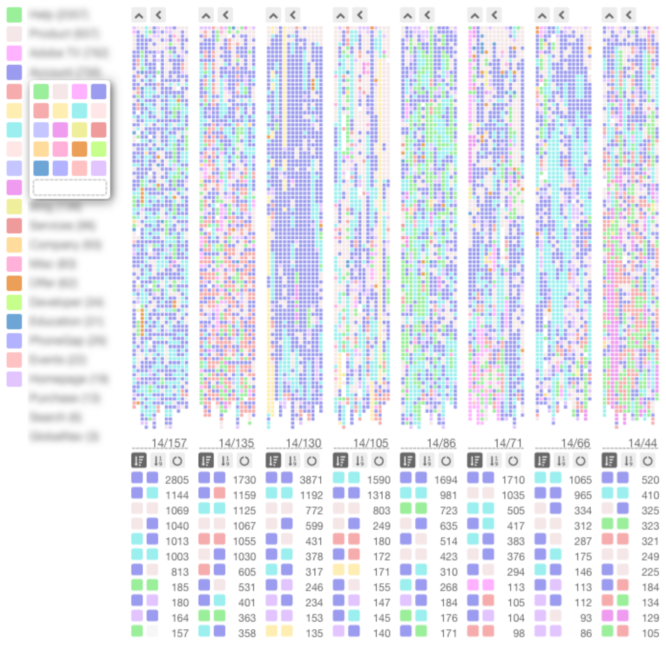


Fig. 5: A design alternative based on sequence clustering. Refer to the supplemental materials for a complete review of the designs we have considered.

the visual complexity. When hovering a single sequence, the detailed event information will be revealed. Using the side panel, we allow the users to change the colors of event categories on-the-fly. Users can even set event categories that are less of concern to transparent to reduce the number of color channels, and the clustering result will be updated accordingly. Furthermore, we allow users to stack the sequences both vertically and horizontally. Vertical stacking helps condensing events of the same category (color), or repeated motifs into super motifs. Horizontal stacking merges sequences consist of the same set of events (a bar chart histogram is used to indicate the number of same sequences). Stacking in both directions can simplify the sequences within each clusters to reduce visual complexity. In addition, we also added an n-gram view to reveal the statistics of transitions between events (bottom part of Figure 5).

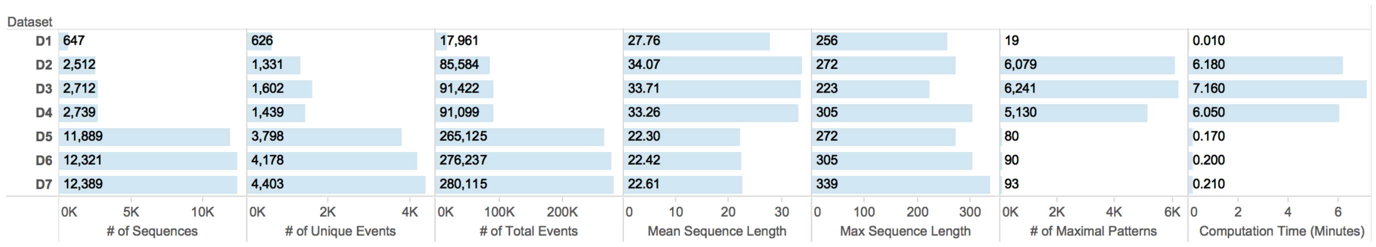
This clustering-based design helps surfacing groups of sequences. However, it is hard to understand how the clusters differ from each other. In addition, it is difficult to provide visual or text summaries for each cluster. The lack of meaningful concepts to characterize these clusters led us to explore other data mining methods.

6.3 Sequential Pattern Mining

Similar to motifs, sequential patterns are ordered list of events that co-occur frequently in a sequence dataset. The main difference is that the events need not be consecutive. Sequential patterns are thus more effective in finding frequent item sets despite the noise and heterogeneity of data. Compared to clustering methods, the parameters and end results of pattern mining algorithms are usually easy to interpret, and users can understand how the extracted patterns are related to raw sequences.

In terms of task support, sequential pattern mining affords presenting and relating information at four granularity levels: patterns, segments, sequences and events (Task 1 and 2). A pattern is a meaningful abstraction of a set of raw sequences. It can be further broken down by the events into pattern segments, an idea inspired by the analysts’ comments on seeing activities happened before and after a motif in the context of original sequences. In addition, if loops are a dominant pattern, they should remain visible after data reduction (Task 3).

In the next three sections, we present details on the pattern mining algorithm we use, the method to prune the pattern space, and design rationales for the interactive visualizations.



Dataset	# of Sequences	# of Unique Events	# of Total Events	Mean Sequence Length	Max Sequence Length	# of Maximal Patterns	Computation Time (Minutes)
D1	647	626	17,961	27.76	256	19	0.010
D2	2,512	1,331	85,584	34.07	272	6,079	6.180
D3	2,712	1,602	91,422	33.71	223	6,241	7.160
D4	2,739	1,439	91,099	33.26	305	5,130	6.050
D5	11,889	3,798	265,125	22.30	272	80	0.170
D6	12,321	4,178	276,237	22.42	305	90	0.200
D7	12,389	4,403	280,115	22.61	339	93	0.210

Fig. 6: Statistics on the datasets we have analyzed and the maximal sequential patterns extracted from these datasets. The patterns are computed on a quad-core 2.7 GHz MacBook Pro (OS X 10.11.5) with per-core 256K L2 caches, shared 6MB L3 cache and 16GB RAM.

7 Mining Sequential Patterns

In this section, we review the concept of sequential pattern mining. We focus on a specific type of sequential patterns: maximal sequential patterns (MSP).

Definition 1. Sequence Dataset: A sequence dataset D is an unordered set of sequences: $D = \{S_1, S_2, ..., S_s\}$

Definition 2. Sequence: A sequence S is an ordered list of events E_i, where i denotes the index of the event in the sequence: $S = [\, E_1, E_2, ...E_n \,]$.

Definition 3. Containment, Super Pattern and Sub Pattern: A sequence $S_a = [\, A_1, A_2, ...A_m \,]$ is *contained* in another sequence $S_b = [\, B_1, B_2, ...B_n \,]$ if there exist integers $1 \le i < j < ... < k \le n$ such that $A_1 = B_i, A_2 = B_j$, ..., and $A_m = B_k$. We denote containment as $S_a \sqsubseteq S_b$. S_b is a *super pattern* of S_a, while S_a is a *sub pattern* of S_b.

For example, consider the sample dataset with three unique events α, β, γ and the four sequences in Table 1. Sequence 2 is contained in sequence 1, and sequence 1 is a super pattern of sequence 2. Sequences 3 and 4 are not contained by any other sequences.

#	Sequence	Metric (number of visitors)
1	$[\, \alpha, \beta, \beta, \gamma, \gamma, \beta, \beta, \gamma, \gamma \,]$	30
2	$[\, \alpha, \gamma, \beta, \gamma, \gamma \,]$	112
3	$[\, \gamma, \alpha, \beta, \gamma, \gamma, \gamma \,]$	57
4	$[\, \beta, \gamma, \beta, \alpha, \gamma \,]$	6

Table 1: A sample clickstream dataset.

Definition 4. Sequential Pattern: A sequential pattern (or "pattern" for short) P is a sequence contained in one or more sequences in the sequence dataset D: $\exists \, \{S_1, S_2, ...S_k\} \subseteq D, k > 0$ such that $P \sqsubseteq S_1, P \sqsubseteq S_2, ..., P \sqsubseteq S_k$. We call each event in a sequential pattern a **Key Event**.

$P = [\, \beta \dashrightarrow \gamma \dashrightarrow \gamma \,]$ is a sequential pattern that summarizes the sequence dataset in Table 1. We use the dashed arrow $\dashrightarrow$ to differentiate sequential patterns from regular event sequences.

Definition 5. Support Set and Support of a Pattern: The *support set* of a pattern P is the set of sequences in a dataset D that contain the pattern: $\text{supset}(P) = \{S | S \in D, P \sqsubseteq S\}$.

The *support* of a pattern P is the percentage of sequences in the dataset that contains the pattern [1]: $\text{supp}(P) = \frac{|\text{supset}(P)|}{|D|} \times 100\%$.

It is rare for a sequential pattern to be contained in *all* of the sequences. In the dataset in Table 1 for example, the support of pattern $[\, \beta \dashrightarrow \gamma \dashrightarrow \gamma \,]$ is 100%, and the support of $[\, \beta \dashrightarrow \gamma \dashrightarrow \gamma \dashrightarrow \gamma \,]$ is 50% (only sequence 1 and 3 contain this pattern). Note that here we are computing the support based on the number of unique sequences that contain the pattern. If we use other metrics associated with the sequences (e.g. number of visitors in Table 1), the support of $[\, \beta \dashrightarrow \gamma \dashrightarrow \gamma \dashrightarrow \gamma \,]$ is 42.4%.

Definition 6. Segment: Given a sequential pattern $P = [\, E_1 \dashrightarrow E_2 \dashrightarrow, ..., E_n \,]$, a segment Seg is a sub-pattern consisting of two adjacent key events in the pattern: $Seg = [E_i \dashrightarrow E_{i+1}]$, $0 \le i \le n-1$. For example, $[\, \beta \dashrightarrow \gamma \,]$ is a segment of $[\, \beta \dashrightarrow \gamma \dashrightarrow \gamma \,]$

Definition 7. Maximal Sequential Pattern: Given a minimum support min_{supp}, a pattern is *maximal* if none of its super patterns has support greater than or equal to min_{supp} [1].

Assume that we want to find sequential patterns with a minimum support of 100% out of the sequences in Table 1, several patterns match this criterion. Out of these, $P = [\, \beta \dashrightarrow \gamma \dashrightarrow \gamma \,]$ is a maximal sequential pattern; $Q = [\, \beta \dashrightarrow \gamma \,]$ is also a matching sequential pattern, but it is not maximal since P contains Q and $\text{supp}(P) = 100\%$.

Many algorithms are available for mining maximal sequential patterns [5, 7, 8, 9, 18, 19, 27]. In our current investigation, we choose the vertical maximal sequential pattern mining algorithm (VMSP) because it does an exhaustive search in the pattern space while maintaining the best performance in terms of computation time [7]. We use the Java implementation of the algorithm in the SPMF library [6].

Given the high cardinality of events in our datasets, we have the option to aggregate events by predefined categories before performing pattern mining. We have experimented with this approach and decided not to aggregate the events. The maximal patterns mined with sequences in full event names manifest greater variety and offer richer information to the analysts.

To extract all the potential sequential patterns of interest in the dataset to the analysts, we want to search for all the maximal patterns with different levels of support. For example, Table 2 shows all the maximal sequential patterns in Table 1 with different supports. We iteratively compute patterns by varying the level of support min_{supp}. Given a predefined range $[x, y]$, and starting from y, we iteratively decrease min_{supp} by a fixed percentage δ. To ensure an exhaustive search, the theoretical setting for the decrement is that $\delta = \frac{1}{|D|} \times 100\%$, where $|D|$ denotes the cardinality of the dataset. In practice, the VMSP algorithm will return patterns with support that is slightly higher than the input min_{supp} parameter. For the datasets we are working with, we find that a decrement of 3% is sufficient to capture all the patterns. We run the VMSP algorithm using this approach on seven clickstream datasets. These datasets are from two different companies and represent data from different time periods and user group. Figure 6 shows the size and characteristics of these datasets.

Besides maximal patterns, other pattern definitions are available, such as frequent patterns [2] and closed patterns [7]. We chose maximal patterns over frequent patterns or closed patterns because we want to avoid presenting too many patterns for analysts to digest, and by definition maximal patterns are a subset of closed patterns, which are in turn a subset of frequent patterns. As Figure 6 shows, clickstream datasets can contain too many patterns even for the strictest definition of maximal patterns.

8 Pruning Sequential Patterns

Depending on the dataset, a given min_{supp} might yield no maximal sequential pattern or multiple ones. After iterating over the support levels and collecting unique sequential patterns, the number of patterns found and the time taken also vary by dataset. Figure 6 presents the statistics on mining MSPs from the seven datasets for supports ranging from 95% to 15% with $\delta = 3\%$. In general, the time taken to mine the pattern is positively correlated with the number of patterns in the dataset.

Support	Maximal Sequential Patterns	Matching Sequence IDs
100%	$[\beta \dashrightarrow \gamma \dashrightarrow \gamma]$	1, 2, 3, 4
75%	$[\gamma \dashrightarrow \beta \dashrightarrow \gamma \dashrightarrow \gamma]$	1, 2, 3
	$[\alpha \dashrightarrow \gamma \dashrightarrow \gamma \dashrightarrow \gamma]$	1, 2, 3
50%	$[\gamma \dashrightarrow \gamma \dashrightarrow \gamma \dashrightarrow \gamma]$	1, 3
	$[\beta \dashrightarrow \gamma \dashrightarrow \beta \dashrightarrow \gamma]$	1, 4
	$[\alpha \dashrightarrow \beta \dashrightarrow \gamma \dashrightarrow \gamma \dashrightarrow \gamma]$	1, 3
	$[\alpha \dashrightarrow \gamma \dashrightarrow \beta \dashrightarrow \gamma \dashrightarrow \gamma]$	1, 2

Table 2: Maximal sequential patterns extracted from Table 1

The VMSP algorithm reduces the sequence length effectively (the maximum pattern length is 8 for the seven datasets in Figure 6), but not necessarily the number of sequences. As shown in Figure 6, the number of extracted patterns is sometimes even greater than the number of sequences. Simply showing patterns with the top k supports is not ideal because the support is only one measure of a pattern's relevance. When we examine the top 10 patterns extracted from each of the datasets, for example, we have observed that many patterns share the same set of unique events and differ only in terms of the number of occurrences and the index of certain events. In addition, many patterns have similar support levels and are contained in a similar set of input sequences.

To prune the pattern space, we use the Jaccard index [10] to measure the similarity between a pair of patterns:

$$J(P,Q) = \frac{|\text{supset}(P) \cap \text{supset}(Q)|}{|\text{supset}(P)| + |\text{supset}(Q)| - |\text{supset}(P) \cap \text{supset}(Q)|}$$

If the support sets of the two patterns have no overlap, $J(P,Q) = 0$; if the two patterns have identical support sets, $J(P,Q) = 1$.

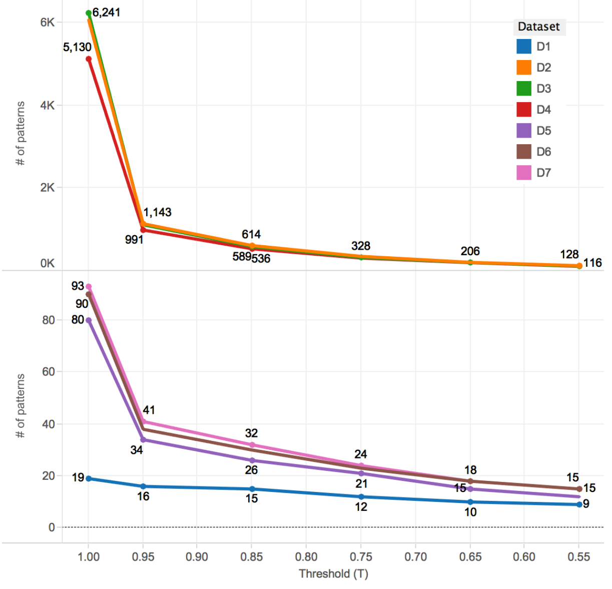


Fig. 7: The number of patterns after pruning with different values of threshold t. The line graphs at the top show the statistics for three datasets with around 6k patterns, the bottom line graphs show the statistics for the other four datasets with smaller number of patterns.

To generate the final pattern set L to be shown to the analysts, we sort the patterns by their supports and pattern length (longest pattern first) and add each pattern P to L if two conditions are met: 1) the Jaccard index between P and each element in L is below a predefined threshold t; 2) the two patterns do not contain the same set of unique events. To efficient compute the Jaccard index, we implemented this approach using bitmap representations with the bitwise AND operator. Figure 7 shows the number of patterns with different values of t for the seven datasets.

9 Visualization and Interaction Design

We now identify the following design goals for the visual interface based on the mined maximal sequential patterns:

1. Visually distinguish sequential patterns from raw sequences: To differentiate patterns from input sequences, we need to design different visual representations for these two types of data.

2. Allow analysts to focus on data of interest: Since the VMSP algorithm can still produce a large number of patterns, and a pattern can represent hundreds of sequences or more, the analysts need to be able to filter out irrelevant data and focus on a particular pattern, a pattern segment or selected sequence segments.

3. Support easy navigation between levels of abstraction: analysts should be able to examine sequential patterns, drill down to detailed sequence and event information, and focus on segments of interest in a unified interface.

The final design consists of two major views (Figure 1): the pattern view displays extracted maximal sequential patterns, and the sequence view shows individual clickstream sequences for detailed analysis. An auxiliary context view provides contextual information on which part of the dataset is the current focus of exploration. The views are coordinated via interaction techniques including overview+detail, sequence alignment and segmentation, and hierarchical pattern mining.

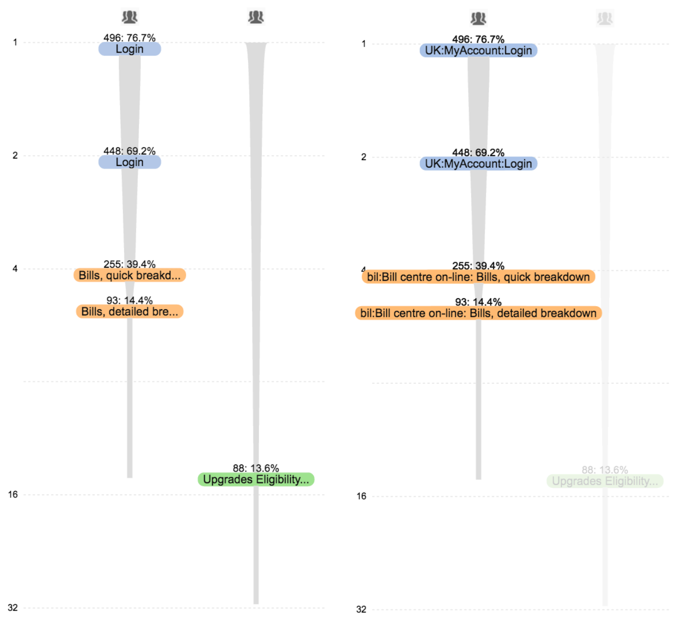
9.1 Pattern View

The pattern view presents sequential patterns extracted from the dataset and pruned using the approach described in Section 8. We display the patterns from left to right, sorted by metrics such as support or pattern length. When the number of patterns is still too large after pruning, users can filter patterns by pattern length and search for patterns containing a particular event.

For each pattern, we lay out the events from top to bottom. The patterns always start at step one, and the length of the patterns represents the average sequence length in the corresponding support sets. For example, in Figure 8a, the support set of the pattern containing only one event "upgrade eligibility" has an average sequence length of 32. For each pattern, the vertical position of each event represents the mean number of clicks the visitors took to reach that event (Figure 8a). Here we see that on average it takes longer for visitors to reach the "upgrade eligibility" page than to reach the billing pages. Since the mean statistics are sensitive to outliers, the analysts can also change the vertical scale to represent other summary statistics, such as the median number of clicks. We have observed, based on many of the datasets we have worked on, that consecutive events in a pattern tend to happen at the beginning of sequences, which results in visual occlusion when we plot the events using a linear scale. As a result, we choose a log scale by default.

The events often have long, hierarchical names such as "bill:Bill centre on-line: Bills, quick breakdown". To avoid visual occlusions that arise due to such long names, we split the labels by delimiters that separate the hierarchical levels and use color to encode the first token, which is usually the event category. The color palette is based on the 20 categorical color scale in D3 [4]. We display the last token as the event name. The analysts can hover a pattern by placing the cursor over the visitors icon (Figure 8a) or select a pattern by clicking on the icon. The other patterns will fade into the background, and the focused pattern will display the full event names (Figure 8b).

For analysts who are familiar with funnel analysis and visualization, each of these patterns resembles a funnel. The only difference is that these patterns are automatically suggested by the tool. According to the analysts, they are interested in traffic statistics for each of the events. We display such information at the top of each event, showing the number and percentage of visitors reaching that event. In general, the number of visitors decreases towards the end of the pattern. The percentage of visitors for the last event in the pattern always denotes the support of the pattern. To provide a sense of continuous visitor flow, we use the width of the line segment connecting adjacent events to represent the change in traffic.



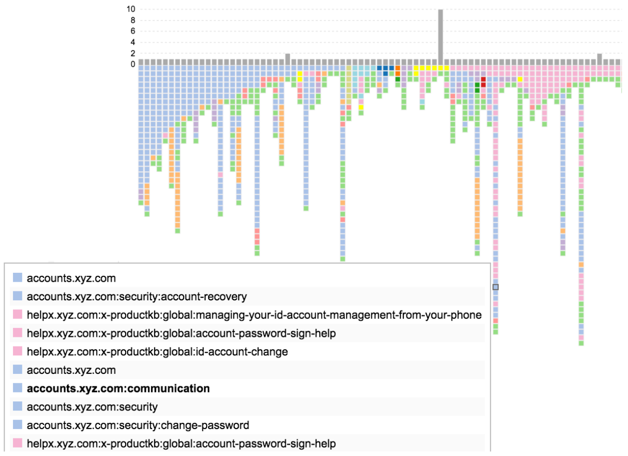
(a) The vertical scale represents the average number of clicks taken by visitors to reach that event, and the size of the links represents the number of visitors transitioning between events.

(b) Hovering over or clicking on the visitor icon on top of a pattern highlights the pattern with complete event names, and the other patterns fade to the background.

Fig. 8: The pattern view showing two maximal sequential patterns. It takes longer for visitors to reach the "upgrade eligibility" page than the billing pages.

9.2 Sequence View

The sequence view provides detailed sequence information for a selected sequential pattern. When users click on the visitor icon at the top of a pattern, the sequence view displays sequences in the pattern's support set. The events in each sequence are laid out from top to bottom, and color coded by a chosen event attribute (e.g. in Figure 9, the color represents event category). For datasets with multivariate events, users can choose which categorical attribute should be encoded using color. When hovering over an event in a sequence, a tooltip will appear, showing the full event names for the adjacent events (Figure 9).



10
8
6
4
2
0
accounts.xyz.com
accounts.xyz.com:security:account-recovery
helpx.xyz.com:x-productkb:global:managing-your-id-account-management-from-your-phone
helpx.xyz.com:x-productkb:global:account-password-sign-help
helpx.xyz.com:x-productkb:global:id-account-change
accounts.xyz.com
accounts.xyz.com:communication
accounts.xyz.com:security
accounts.xyz.com:security:change-password
helpx.xyz.com:x-productkb:global:account-password-sign-help

Fig. 9: The sequence view shows individual sequences laid out from top to bottom. The top bar chart shows the aggregated metrics for each sequence. Hovering over an event displays the event names.

Each sequence may have a number of metrics such as the number of visitors following that path. We use a gray bar chart on top to show such information. A design alternative is to encode such quantitative metrics as the width of the sequence. Such an approach tends to give visual emphasis to the most frequent sequences, and less frequent sequences may not be easily identifiable. While frequent sequences may be interesting, outliers and less frequent sequences can be more important in actual analysis. As a result, we choose to encode sequence metrics as a bar chart in order to give every sequence equal emphasis.

The order of the sequences influences the visual pattern emerging from the display. We sort the sequences such that events belonging to the same category are placed next to each other (Figure 9). Upon closer inspection, the result of this sorting algorithm yields a visualization resembling an icicle plot [15], which has been used in event sequence visualizations [35, 36]. The main difference between the sequence view and the icicle plot is that we maintain the individuality of the sequences and do not merge the events at each step. Doing so gives us the flexibility to show detailed event names on demand, and to support alternative sorting metrics such as sequence length or number of visitors. We have also considered alternative methods to visualize the sequences such as the Sankey diagram or the MatrixWave design [38]. These methods do not provide similar levels of scalability and flexibility.

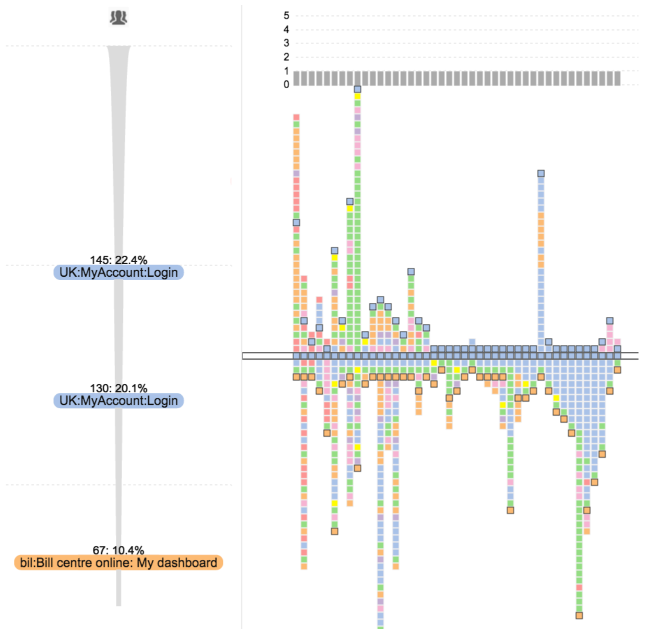
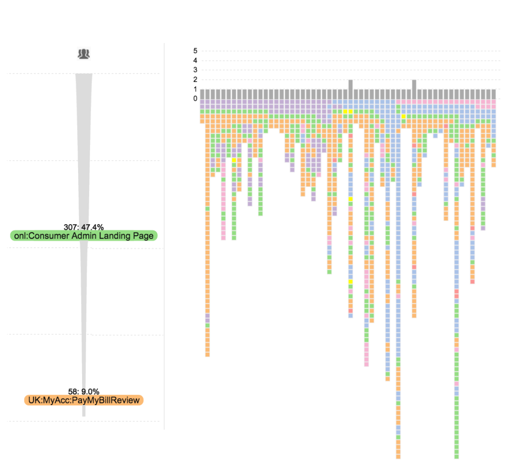


Fig. 10: Clicking on an event in the pattern view will align the clickstreams in the sequence view by that event. In this visualization, the clickstreams are aligned by the second "UK:MyAccount:Login" event in the pattern on the left.

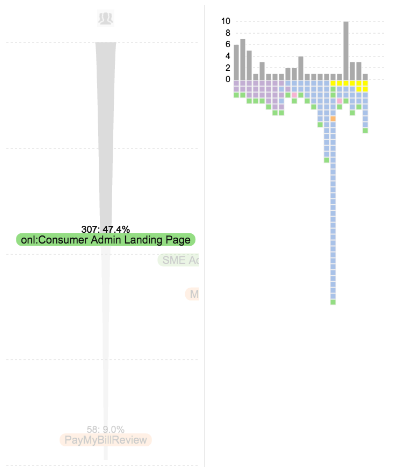
9.3 Highlight Key Events and Align Sequences by Event

By default, the sequences align at the top and start at the first step in the Sequence View. It is not visually clear to the analysts how the sequences in a pattern's support set manifest the pattern. We design two features to allow the analysts to understand the relationships between the pattern and the sequences better. First, users can turn on event highlighting, which renders dark gray borders around each key event in the sequences (Figure 10). Second, users can click on any of the events in a pattern, the sequences will be aligned by the chosen event, allowing users to examine the leading and trailing events (Figure 10). We apply animated transitions to show how sequences line up by different key events.

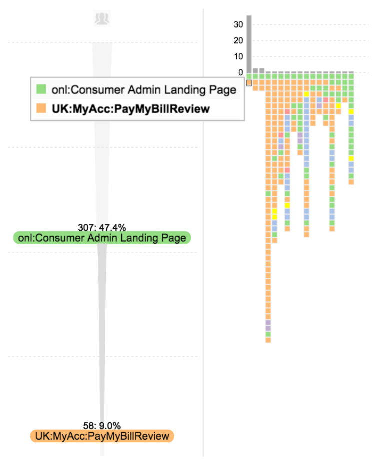
It is possible for a pattern to contain multiple occurrences of the same event, and a sequence in the support set can have many occurrences of that event as well. When searching for a key event E_k in a sequence, we first look for the indices of the key events preceding E_k, and return the position of the first occurrence of E_k after those indices.



(a) A pattern with two events ("onl:Consumer Admin Landing Page" and "UK:MyAcc:PayMyBillReview") and the clickstreams in its support set.



(b) Selecting the first pattern segment shows the sequence segments from the beginning of the clickstreams to the "onl:Consumer Admin Landing Page".



(c) Selecting the second pattern segment shows the sequence segments from the "onl:Consumer Admin Landing Page" to the "UK:MyAcc:PayMyBillReview" page.

Fig. 11: Analysts select pattern segments to focus on parts of the visitors' journey.

9.4 Focus on Sequence Segments

The support set of a pattern can still contain hundreds or even thousands of sequences. The analysts are unlikely to browse every sequence in this case. In order to make sequence analysis easier, we allow users to break down sequences into segments by events. Figure 11 shows how analyzing sequence segments can reveal more insights with less overwhelming visualizations. When users click on a segment linking two adjacent events in a pattern, we truncate the sequences in the support set using the two events and aggregates the truncated sequences. The truncated sequences are often more compact and revealing. For example, Figure 11c shows that most people go to the "PayMyBillsReview" page directly from the "Consumer Admin Landing Page". This insight is not obvious from Figure 11a.

9.5 Hierarchical Pattern Mining

When the number of sequences in a pattern's support set is large, truncating sequences into segments using the pattern's events could help. It also makes sense to perform hierarchical pattern mining on sequence segments as well. The analysts can right click on any of the segments in the Pattern View to invoke a context menu, and run the Vertical Maximal Sequential Pattern (VMSP) mining algorithm on the corresponding sequence segments (Figure 12a). The contextual view provides cues to indicate which part of sequence dataset is the current focus of investigation (Figure 12b).

Hierarchical pattern mining can also take place based on a user-defined funnel. As discussed in Section 3, one commonly used analytic tool is to define events in the funnel and observe how traffic falls out the funnel. In our design, the analysts can define a funnel, and the context view visualizes the number of visitors reaching each key event step in the funnel (Figure 1). They can further select funnel segments and extract maximal sequential patterns to drill down into parts of the data (Figure 1).

The context view also serves as a navigation map for users to return to an overview of sequential patterns or to continue pattern mining for a different segment of the pattern. Clicking on a segment in the context view will update the pattern view accordingly to display the corresponding sequential patterns.

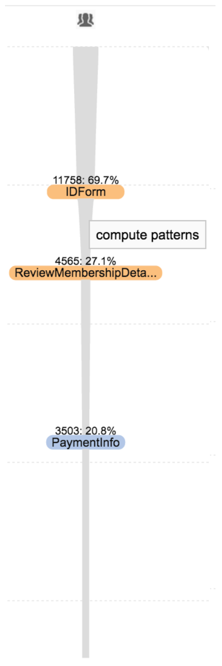
10 Evaluation

10.1 Analysis Scenario

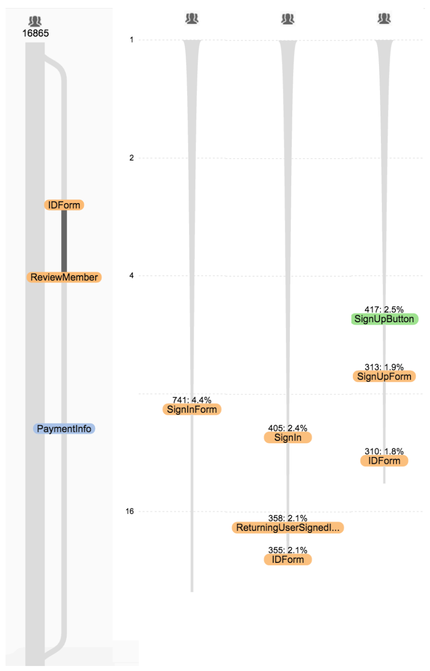
We have applied the methods discussed in Section 8 and Section 9 to real-world clickstream datasets from two companies with different time windows. To illustrate how these methods are being used in actual analysis, we present an exemplary scenario in this section. The dataset comes from a telecommunications company that offers mobile phones and service plans, and contains 1361 visit sessions. The sessions are aggregated into 647 sequences based on page names, with 626 unique events and 17961 events in total. The longest sequence has 256 events, and the average number of events in the sequences is 27.8.

To identify key customer journeys (Task 1), we run the VMSP algorithm on this dataset produces 19 patterns, and applying the pruning method described in Section 8 with threshold $t = 0.65$ gives us 10 final patterns. These patterns represent common customer journeys taken by the visitors. Browsing in the pattern view, the analyst sees that, for example, 39.4% of the visitors look at quick breakdown of their bills after logging in, and 14.4% continue to dive into the detailed billing information.

The analyst is interested in the recent promotion on new data plans,



(a) Mining patterns on a pattern segment through a context menu.



(b) The context view highlights the pattern segment being examined at the moment. The pattern view shows maximal patterns for the selected pattern segment.

Fig. 12: Hierarchical pattern mining allows analysts to drill down into pattern segments of interest.

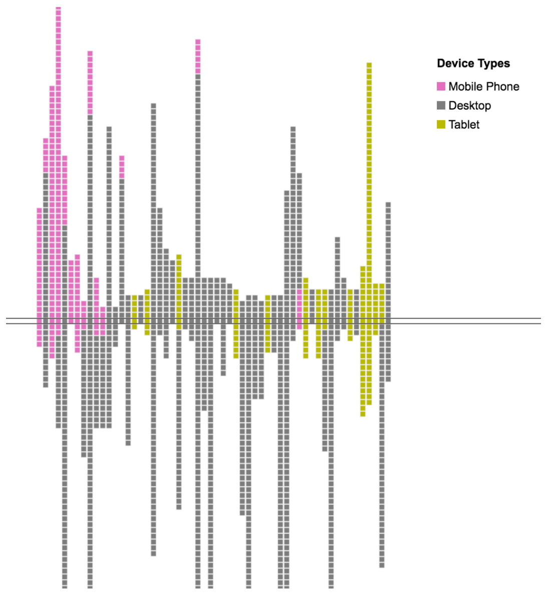


Fig. 13: Changing event color encoding from event category to device type reveals insights on how visitors browse website pages using different devices.

so she looks for patterns containing the "upgrade eligibility" page. One pattern shows that 8.7% of the visitors check whether they are eligible for upgrades after logging in. To examine it more closely, the analyst clicks on the visitor icon to select the pattern, and the sequence view displays the individual clickstreams (Task 2). She aligns the sequences by the "upgrade eligibility" page to understand visitor behavior before and after this milestone. Certain visitors are experiencing problems with login and some spent as many as more than 30 steps in the login process (Task 4). Many visitors were browsing phone and plans after the eligibility page.

To get a better idea of how visitors spent their time in between the "login" page and the "upgrade eligibility" page, the analyst right clicks on that pattern segment and performs pattern mining on the corresponding sequence segments. About 4% of the visitors stopped at the "consumer admin page", which contained links to the "upgrade eligibility" page. Interestingly 1% of the visitors went through the NBA channel page under the company's cable department. The analyst notes that it may be worthwhile to compare customers who subscribe to the NBA channel against those who do not (Task 2).

Finally, the analyst would like to understand the influence of device type on visitor behavior. She goes back to the "login" - "upgrade eligibility" pattern and switches the color encoding in the sequence view from "page category" to "device type" (Task 3). The sequence view now updates the event colors accordingly (Figure 13). She sees that visitors using mobile phones tend to have shorter sequences, and all the longer sequences where people were browsing phones and plans are from desktop devices. This can be an indication that the mobile sites for products and services need improved designs to provide better browsing experiences.

10.2 User Feedback

We implemented the sequence mining and visualization methods in an initial prototype and demonstrated the tool to the analysts using a clickstream dataset from a software firm. The analysts made positive comments, noting the power of sequence mining combined with interactive visualization for exploratory analysis. According to the analysts, they considered the sequential patterns as visual indices into the clickstreams. Before this tool, there was no way to get a high-level view of the dataset, neither was drilling down into individual sequences available. Our tool provides visual cues for them to examine high-level patterns and focus on sequences they care about. One analyst said, "*I think this is a great step towards pathing. We have been trying to figure out pathing and how the user goes throughout the site and how they drop off. This will help us.*" Another made similar remarks: "*Awesome. A difficult task made much easier. Key customer journeys within the funnel a major improvement of the current funnel presentation. This would drastically help answer many questions around UX and navigation decisions.*"

The reactions on the sequence view were mixed. Some analysts valued the ability to see individual sequences aligned by a key event: "*[the sequence view] was one of the features that was appealing since you can see clear volume paths that take more steps or less steps to get to a certain point*". Others felt the sequence view to be overwhelming, although it does not invalidate the usefulness of the tool: "*Despite the somewhat overwhelming graph [the sequence view] presented, the insight is significant.*"

Some analysts requested the capability to define sequences. A series of events can be grouped, divided or concatenated in different ways to form sequences. For example, the analysts may want to break down a sequence from a visitor into sessions; or to stitch together sequences by a user ID. It is interesting to understand how different sequence definitions influence the analysis outcomes. Another frequently requested features is to group and compare sequences by traffic sources (e.g. search, referral, or social media).

10.3 Limitations and Future Work

Analysts' feedback as well as our own experience suggest that the proposed approach also has several limitations. First, the size of clickstream dataset used in Section 10.1 is very modest. Depending on the time frame and filters used in analysis, real-world datasets can contain up to hundreds of thousands unique sequences. The amount of time taken to compute the patterns depends on the number of patterns in a dataset (Figure 6 shows the time to compute MSPs for the sample datasets we have examined). For large datasets, the computation time goes into the range of minutes or even hours. Future work along this thread includes exploring the possibility of parallelizing pattern computation and progressive pattern computation [31] to reduce the latency and improve user experience.

With increased data size, we also need to examine the impact of pattern pruning methods on the quality of final patterns being presented. In this paper, we described a simple approach based on mutual information between patterns. It remains an unexplored area to define interestingness or usefulness measures for sequential patterns and to automate the process of selecting interesting patterns for visualization.

Finally, the scalability of the visualization interface can be improved. The context view becomes crowded if we repeatedly perform hierarchical pattern mining on pattern. The sequence view also becomes harder to use with larger datasets. We plan to address these issues along two directions: 1) replace the sequence view with higher-level summaries of event occurrences such as bar charts showing the number of events per category, and show individual sequences on demand; 2) augment the pattern view with interactive techniques such as semantic zooming and expansion, so that the analysts can view the raw sequences in the context of high-level patterns.

11 Conclusion

In this paper, we review design rationales in choosing data mining methods and visual representations for clickstream exploration across multiple levels of granularity. Using sequential mining as the primary data reduction approach, we present an analytic pipeline consisting of three stages: pattern mining, pattern pruning, and coordinated exploration. We propose visualization designs for the sequential patterns and raw sequences, and design interaction techniques to help analysts make sense of the patterns and sequences. The feedback from professional analysts is positive, and confirms that our design addresses the important analytic tasks in their work.

References

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In *Data Engineering, 1995. Proceedings of the Eleventh International Conference on*, pages 3–14. IEEE, 1995.

[2] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu. Sequential pattern mining using a bitmap representation. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 429–435. ACM, 2002.

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

[4] M. Bostock, V. Ogievetsky, and J. Heer. D3: Data-driven documents. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2011.

[5] M. El-Hajj and O. R. Zaiane. Parallel leap: large-scale maximal pattern mining in a distributed environment. In *Parallel and Distributed Systems, 2006. ICPADS 2006. 12th International Conference on*, volume 1, pages 8–pp. IEEE, 2006.

[6] P. Fournier-Viger, A. Gomariz, T. Gueniche, A. Soltani, C.-W. Wu, and V. S. Tseng. Spmf: a java open-source pattern mining library. *The Journal of Machine Learning Research*, 15(1):3389–3393, 2014.

[7] P. Fournier-Viger, C.-W. Wu, A. Gomariz, and V. S. Tseng. VMSP: Efficient vertical mining of maximal sequential patterns. In *Advances in Artificial Intelligence*, pages 83–94. Springer, 2014.

[8] P. Fournier-Viger, C.-W. Wu, and V. S. Tseng. Mining maximal sequential patterns without candidate maintenance. In *Advanced Data Mining and Applications*, pages 169–180. Springer, 2013.

[9] E.-Z. Guan, X.-Y. Chang, Z. Wang, and C.-G. Zhou. Mining maximal sequential patterns. In *Neural Networks and Brain, 2005. ICNN&B'05. International Conference on*, volume 1, pages 525–528. IEEE, 2005.

[10] P. Jaccard. The distribution of the flora in the alpine zone. *New phytologist*, 11(2):37–50, 1912.

[11] D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, and G. Melançon. Visual analytics: Definition, process, and challenges. In *Information visualization*, pages 154–175. Springer, 2008.

[12] D. A. Keim. Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on visualization and computer graphics*, 6(1):59–78, 2000.

[13] D. A. Keim, J. Schneidewind, and M. Sips. Fp-viz: Visual frequent pattern mining. 2005.

[14] J. Krause, A. Perer, and H. Stavropoulos. Supporting iterative cohort construction with visual temporal queries. *Visualization and Computer Graphics, IEEE Transactions on*, 22(1):91–100, 2016.

[15] J. B. Kruskal and J. M. Landwehr. Icicle plots: Better displays for hierarchical clustering. *The American Statistician*, 37(2):162–168, 1983.

[16] H. Lam, D. Russell, D. Tang, and T. Munzner. Session viewer: Visual exploratory analysis of web session logs. In *Visual Analytics Science and Technology, 2007. VAST 2007. IEEE Symposium on*, pages 147–154, Oct 2007.

[17] Z. Liu, B. Lee, S. Kandula, and R. Mahajan. Netclinic: Interactive visualization to enhance automated fault diagnosis in enterprise networks. In *Visual Analytics Science and Technology (VAST), 2010 IEEE Symposium on*, pages 131–138. IEEE, 2010.

[18] C. Luo and S. M. Chung. Efficient mining of maximal sequential patterns using multiple samples. In *SDM*, pages 415–426. SIAM, 2005.

[19] C. Luo and S. M. Chung. Parallel mining of maximal sequential patterns using multiple samples. *The Journal of Supercomputing*, 59(2):852–881, 2012.

[20] E. Maguire, P. Rocca-Serra, S.-A. Sansone, J. Davies, and M. Chen. Visual compression of workflow visualizations with automated detection of macro motifs. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2576–2585, 2013.

[21] S. Malik, F. Du, M. Monroe, E. Onukwugha, C. Plaisant, and B. Shneiderman. Cohort comparison of event sequences with balanced integration of visual analytics and statistics. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, IUI '15, pages 38–49, New York, NY, USA, 2015. ACM.

[22] S. Malik, B. Shneiderman, F. Du, C. Plaisant, and M. Bjarnadottir. High-volume hypothesis testing: Systematic exploration of event sequence comparisons. *ACM Transactions on Interactive Intelligent Systems*, 6(1):9:1–9:23, Mar. 2016.

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

[24] D. Oelke, D. Spretke, A. Stoffel, and D. A. Keim. Visual readability analysis: How to make your writings easier to read. *IEEE Transactions on Visualization and Computer Graphics*, 18(5):662–674, 2012.

[25] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. Mining sequential patterns by pattern-growth: The prefixspan approach. *Knowledge and Data Engineering, IEEE Transactions on*, 16(11):1424–1440, 2004.

[26] A. Perer and F. Wang. Frequence: interactive mining and visualization of temporal frequent event sequences. In *Proceedings of the 19th international conference on Intelligent User Interfaces*, pages 153–162. ACM, 2014.

[27] C. Raissi, P. Poncelet, and M. Teisseire. SPEED: Mining maximal sequential patterns over data streams. In *International Conference on Intelligent Systems-ICIS*, pages 1–8. IEEE, 2006.

[28] P. Riehmann, M. Hanfler, and B. Froehlich. Interactive sankey diagrams. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pages 233–240. IEEE, 2005.

[29] M. Rosvall and C. T. Bergstrom. Mapping change in large networks. *PloS one*, 5(1):e8694, 2010.

[30] J. Stasko and E. Zhang. Focus+ context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations. In *Information Visualization, 2000. InfoVis 2000. IEEE Symposium on*, pages 57–65. IEEE, 2000.

[31] C. D. Stolper, A. Perer, and D. Gotz. Progressive visual analytics: User-driven visual exploration of in-progress analytics. *Visualization and Computer Graphics, IEEE Transactions on*, 20(12):1653–1662, 2014.

[32] K. Vrotsou and A. Nordman. Interactive visual sequence mining based on pattern-growth. In *Visual Analytics Science and Technology (VAST), 2014 IEEE Conference on*, pages 285–286. IEEE, 2014.

[33] G. Wang, X. Zhang, S. Tang, H. Zheng, and B. Y. Zhao. Unsupervised clickstream clustering for user behavior analysis. In *SIGCHI Conference on Human Factors in Computing Systems*, 2016.

[34] J. Wei, Z. Shen, N. Sundaresan, and K.-L. Ma. Visual cluster exploration of web clickstream data. In *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pages 3–12. IEEE, 2012.

[35] 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 *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1747–1756. ACM, 2011.

[36] K. Wongsuphasawat and J. Lin. Using visualizations to monitor changes and harvest insights from a global-scale logging infrastructure at twitter. In *Visual Analytics Science and Technology (VAST), 2014 IEEE Conference on*, pages 113–122. IEEE, 2014.

[37] E. Zgraggen, S. M. Drucker, and D. Fisher. (s—qu)eries: Visual regular expressions for querying and exploring event sequences. *Proceedings of CHI 2015*, 2015.

[38] J. Zhao, Z. Liu, M. Dontcheva, A. Hertzmann, and A. Wilson. Matrixwave: Visual comparison of event sequence data. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 259–268. ACM, 2015.