Exploring Multivariate Event Sequences using Rules, Aggregations, and Selections

Bram C.M. Cappers and Jarke J. van Wijk

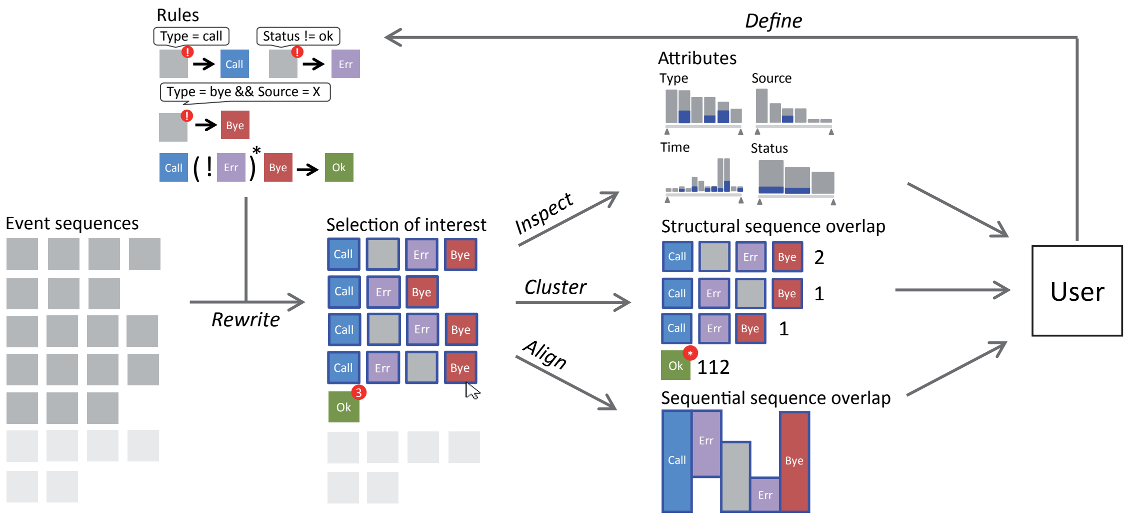


Fig. 1. The Eventpad dataflow model for multivariate event sequence exploration: Users construct rules to visually encode events of interest using attributes and domain knowledge. Exploration of rewritten events is achieved by inspecting overlap in multivariate data of selections of interest. Clustering sequences based on their visual encoding enables users to discover structural overlap between sequences. Overlap between sequentially different sequences is discovered using multiple sequence alignment. New insights can directly be incorporated in the analysis by storing selections of interest and defining new rules throughout exploration.

Abstract— Multivariate event sequences are ubiquitous: travel history, telecommunication conversations, and server logs are some examples. Besides standard properties such as type and timestamp, events often have other associated multivariate data. Current exploration and analysis methods either focus on the temporal analysis of a single attribute or the structural analysis of the multivariate data only. We present an approach where users can explore event sequences at multivariate and sequential level simultaneously by interactively defining a set of rewrite rules using multivariate regular expressions. Users can store resulting patterns as new types of events or attributes to interactively enrich or simplify event sequences for further investigation. In Eventpad we provide a bottom-up glyph-oriented approach for multivariate event sequence analysis by searching, clustering, and aligning them according to newly defined domain specific properties. We illustrate the effectiveness of our approach with real-world data sets including telecommunication traffic and hospital treatments.

Index Terms—Event Visualization, Multivariate Events, Regular Expressions, Sequence Alignment, Interaction

1 Introduction

Many domains nowadays try to gain insight in complex phenomena by logging their behavior. Telecom companies for instance analyze their communication networks for the presence of fraud, hospitals analyze patient treatments to discover bottlenecks in the process, and companies study their work flows to improve customer satisfaction. The common ground here is that domains are interested in the analysis of *sequences* (e.g., phone calls, treatments, work flows) in their system by recording *events*. Without loss of generality, we define a *sequence* (a.k.a. trace, record, session, case, or conversation) as a series of events that have the same *sequence_id*. Besides their type and temporal information, events often have more associated information (e.g., status code, source, length etc.) depending on the domain. In addition, the number of events in real-world data is typically in the order of millions and more.

Multivariate event sequence exploration is still a challenge due to size and variety. Current methods often limit the analysis of event sequences to a single attribute without considering other multivariate event properties in the definition of patterns. We believe however that for root-cause analysis of anomalous sequences the two should be explored simultaneously, since values in multivariate data are often crucial to understand patterns in sequences and vice versa. For example, although sequences of requesting, accessing, and modifying a file in general are not suspicious, they can be considered malicious when they are requested by a particular group of users, the type of file is invalid, and/or one or more authentication errors occurred throughout the execution. For this we need to be able to incorporate multivariate data of events and sequences in our sequential analysis. The observation for instance that all request events in anomalous sequences share particular characteristics in one or more attributes can help analysts to gain insight in the underlying problem.

To enable simultaneous exploration of attributes and event sequences, our exploration method focuses on a user-oriented approach where the

• *Bram C.M. Cappers is with Eindhoven University of Technology. E-mail: b.c.m.cappers@tue.nl*
• *Jarke J. van Wijk is with Eindhoven University of Technology. E-mail: j.j.v.wijk@tue.nl*

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

inspection of selections of interest, creation of rules and reinterpretation of event sequences according to these rules play a central role. To support this flexibility we introduce Eventpad, a notepad editor for multivariate event data. More specifically, our main contributions are:

- an exploration method that enables users to simultaneously explore and analyze multivariate event sequences on both multivariate data and sequential level, using
- a glyph-oriented visual query interface to define higher level patterns using *multivariate* regular expressions,
- a uniform interaction scheme for the creation and modification of selections of interest across events and attributes, and
- a tightly coupled dual view for the discovery of overlap and anomalies between event sequences through interactive multiple sequence alignment and sequence reordering.

The paper is structured as follows. First, related work is discussed in Section 2. Next, we describe our data model and approach to multivariate event sequence exploration in Section 3. Sections 4, 5, and 6 present an overview of the system and discuss design decisions with respect to visualization, data manipulation, and interaction. In Sections 7 and 8 we provide two example explorations on real-world data sets and discuss the limitations of the approach. Finally, conclusions and future work are presented in Section 9.

2 Related Work

Event sequence exploration is an extensively studied topic covering a wide range of visualization techniques. The most well-known method to visualize event sequences is by representing events as glyphs based on their type and point (or interval) in time [8, 11, 12, 22]. To enable exploration in event sequence analysis across different tasks [7], most systems enable users to perform operations on events and sequences such as grouping, aligning, searching, sorting, and filtering [46].

Event sequence exploration methods can be grouped into two main categories: exploration through *overview* or through pattern *searching*.

2.1 Event Overview

An overview is often obtained by showing events using icicle plots [4, 24, 47], state diagrams [48], pixel maps [9], "piano roll" glyph displays [12], or networks [36]. As opposed to the visualization of all event sequences, other approaches focus on the visualization of similarities [30] or differences between sequences, such as MatrixWave [50]. In order to cope with large data volumes, many overview techniques provide filtering and clustering capabilities to reduce visual complexity.

A common approach for event visualizations is to visually encode events according to their type. In case of a large number of event types, this encoding is limited to the most frequent types in the data. Event types are important, but for detailed analysis often multiple attributes have to be taken into account to highlight relevant events. Hence, we aim for an alternative approach where we, similar to Tominski [41] and Bernard et al. [5], enable users to visually annotate and simplify [19,25] the data that at that moment are relevant for their analysis. To support this we enable users to specify *rules*.

2.2 Event Searching

Various visual *search* interfaces have been developed for the construction of time-interval based queries [14, 35], regular expressions [3], analysis of cohort selections [23], and web session logs [26]. A query interface closest to Eventpad is (s|qu)eries [49]. The (s|qu)eries system enables users to visually construct regular expressions on multivariate data associated to events. Users can drag and drop multivariate constraints as blocks in a node-link diagram to build their query of interest. More complex regular expressions can be obtained by adding operators to the blocks and connecting them sequentially via edges. (s|qu)eries is a query-driven system that is effective for finding known patterns of interest. For the specification of queries however, users need to be well aware of the available attribute space, since (besides searching) there is no other support for event sequence exploration and attribute analysis.

2.3 Event Exploration

An exploration method both supporting overview and search mechanisms and close to our technique is EventFlow [33]. EventFlow is an extensive tool for the exploration and analysis of large event collections. Custom interfaces are provided to intuitively search, filter, categorize, align, and simplify [34] events based on type, timestamps, and time intervals. Although the system provides extensive support for event analysis by their type and temporal information, Monroe et al. indicate that the system provides little support for the analysis of additional multivariate data [33]. In addition, they also indicate that comprehensive support across event sequences and attributes is key for temporal event sequence analysis tools to be used in practice. Other techniques try to overcome the problem of multivariate analysis by considering the attributes together as one event type [18]. This approach however does not scale well for high-dimensional data.

Another system for the simultaneous analysis of temporal patterns and multivariate data is ClickStreamVis by Liu et al. [28]. ClickStreamVis enables the analysis of multivariate data in event sequences by extracting frequent sequential patterns from the data using pattern mining techniques. As opposed to existing techniques, ClickStreamVis tries to obtain higher granularity levels by automatically extracting (sub)sequences of interest using motif analysis and various clustering techniques. Unique event sequences are visualized in an icicle plot along with their frequencies. Analysts can align sequences in the visualization according to events in the mined patterns to discover areas of interest. Liu et al. already indicated that their sequence view does not scale well due to the wide variety in large event sequence logs and lack of semantic zooming in their tooling. Furthermore, the computation of mining and pruning maximal sequence patterns can take minutes for relatively small data sets. In Eventpad we tackle wide diversity between event sequences by enabling users to filter or simplify sequences using regular expressions [21]. In addition, users are enabled to store intermediate selections of interest for further investigation. Since the evaluation of a regular expression is linear in the size of the input and its corresponding deterministic finite automaton [2], the technique easily scales to the analysis of hundreds of thousands of events.

In summary, current methods either focus on the sequential analysis of univariate data or the structural analysis of multivariate data. Systems that do take multivariate properties into account during analysis either focus on obtaining an overview or providing search capabilities to explore the data. Currently, no system provides users the ability to both explore and search event sequences by combining temporal patterns and multivariate data in one interface and query language. In addition, no system enables users to interactively explore overlap between sequences of interest alongside attributes using multiple sequence alignment and rule-based event rewriting.

3 Exploration

The size of and variety in large event sequence logs makes it difficult to gain insight in the behavior of the underlying system. Users are not only interested in the existence of particular sequences, but also want to understand what might have caused them. In order to do so, they need to be able to discover which multivariate event properties in these sequences distinguish them from the rest. Since higher level concepts such as a "bad login attempt" or "successful phone call" are typically not captured in low level events, users need to be enabled to enrich their data with these concepts to find the sequences they are interested in. In summary, we need a scalable interactive method to

- inspect event sequence orderings of interest alongside their associated multivariate data,
- assist users in finding and defining sequences of interest, while
- staying aware of high-level phenomena in sequence collections through overview.

In order to keep this method simple and scalable for the analysis of large event logs, we choose for a bottom-up exploration approach

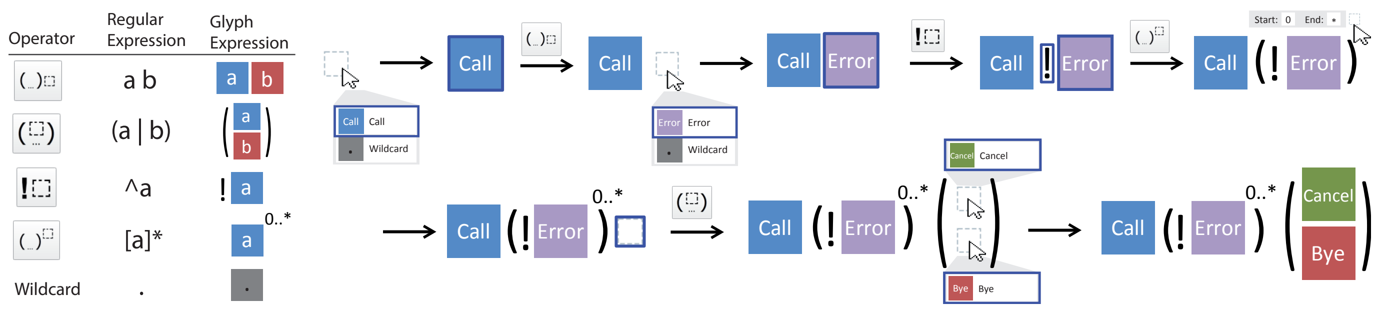


Fig. 2. Incremental construction of regular expressions with predicate logic. The example query involves two different attributes and returns all phone calls starting with a CALL and ending with a BYE or CANCEL without 0 or more error messages in between. Operators can be applied over multiple glyphs by first selecting them through range selection.

where event sequences are represented by series of glyphs and high-level overviews are obtained by finding and summarizing sequences based on user-defined patterns of interest. To tackle the variety in event logs, we consider the problem at three levels, namely the reduction of complexity:

- within event sequences,
- between event sequences, and
- inside multivariate data (of events of interest).

For this we use three concepts, namely

- rules,
- pattern aggregation, and
- selections.

Rules enable users to simplify and visually encode event sequences using glyphs and regular expressions. Users can apply *pattern aggregation* to discover overlap between *sequentially* similar but *structurally* different sequences by summarizing them using clustering and alignment techniques. The creation of event *selections* of interest enables users to focus on parts of the event sequences that are relevant for their investigation. Figure 1 shows a schematic overview of how these concepts are used in Eventpad.

In terms of the strategy guidelines of Du et al. [11], Eventpad enables users to extract records and categories (S1,S2), identify features of sequences (S3) or alignments (S4) with the ability to group events by custom defined categories (S9) and coalescing event sequences into new events (S10, S12). Newly defined search patterns can be stored and applied to larger data sets (S14). For a demonstration of the method in practice, we refer to the supplementary video[1].

4 Rules

Text editors typically enable text filtering, highlighting, and compression by providing search and replace functionality through *regular expressions*. Regular expressions are the de facto standard in industry systems such as Elasticsearch [17], Logstash [42], or `grep` [20] for efficiently finding (and replacing) character sequences in text according to search patterns. Traditional regular expression languages only operate on univariate data such as plain text. We describe how we enable multivariate event sequence analysis by extending regular expressions with predicate logic to support multiple attributes.

4.1 Formal theory

A traditional regular expression consists of *symbols* (i.e., text) and *operators*. In order to extend regular expressions to support multivariate data, we define two types of events, namely *micro* and *macro* events.

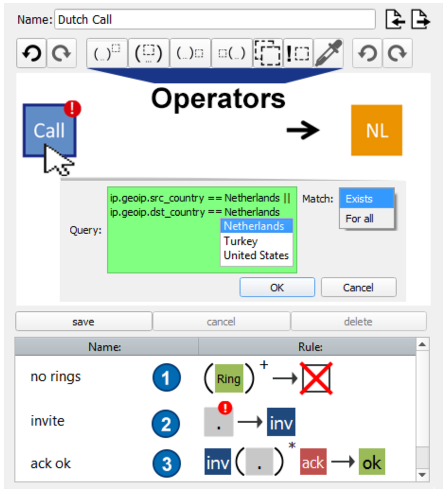
A microevent e has (attribute,value) pairs $(a,v) \in A \times V$ where v can represent numerical ranges, strings, or boolean values. In addition, a microevent has a `sequence_id`. We model a macroevent $e' = \langle L, ES \rangle$ where L is a list of labels and ES a set of microevents. Initially we assume that every microevent e is contained in a (default) macroevent e' with $L = [\text{"Gray"}]$ and $ES = \{e\}$. We can now model an event sequence as a series of macroevents with the same `sequence_id`.

[1]Video: `https://www.youtube.com/watch?v=2DWVW-vLN8Q`

Our extended regular expression language consists of *predicates* and *operators*. A predicate is a boolean expression B over label(s) in $\mathscr{L}$ and/or attribute(s) and value(s) in A and V. A macroevent m satisfies B if and only if **all** microevents in m satisfy B. Alternatively, one can require **at least one** microevent to satisfy B. We refer to this as *maximal* versus *minimal* matching respectively.

A rule is of the form $\alpha \rightarrow l$ where α is a regular expression and l is a label. Operationally, a rule is fired if an event sequence s in the data set can match α. This results in the replacement of s by a new macroevent $c = \langle L', ES' \rangle$ where $L' = [l]$ and ES' is the of union all microevents in s. In case of one-to-one mappings, labels in s are prepended to L'. This enables users to reason about multiple labels when specifying queries (more about this in Section 4.3.)

In contrast to traditional regex, we have to ensure that events before and after the application of a rule remain the same. To keep this mapping simple and intuitive, we limit a rule to the creation of one macroevent. For more details, we refer to the supplementary material.



Query: ip.geoip.src_country == Netherlands || ip.geoip.dst_country == Netherlands
Netherlands
Turkey
United States

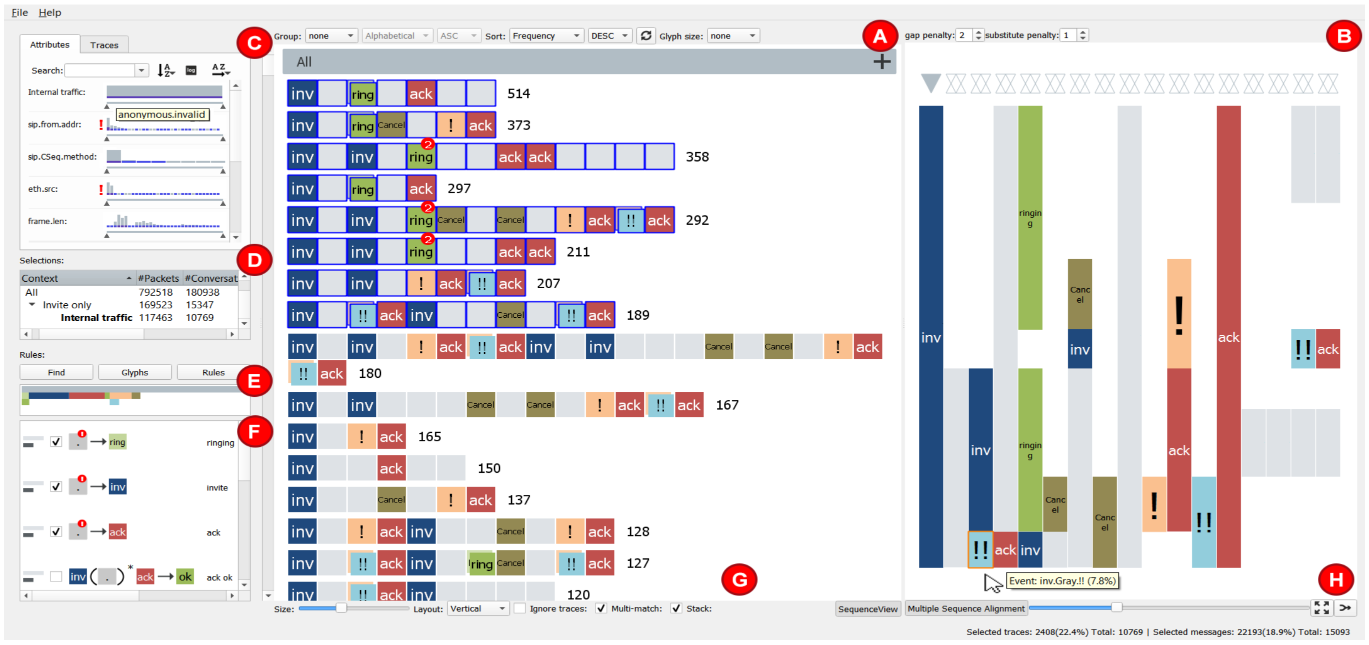
Name:		Rule:
no rings	1	(Ring)+ → ☒
invite	2	. → inv
ack ok	3	inv (.)* ack → ok

Fig. 3. Query interface for the construction of rules. Boolean constraints can be added to a glyph using a textual interface with autocompletion and query validity checks. The pipet enables users to copy/paste glyphs that are present in the sequence view in the query interface. Glyphs with an exclamation mark have one or more attribute constraints.

4.2 Rule visualization

To keep our search and replace interface close to traditional text rewriting, we chose to visually represent macroevent labels as glyphs. Initially every macroevent is represented by a gray glyph.

The sequence view (Figure 4A) visualizes event sequences as series of glyphs, starting every sequence on a new row. Event sequences that do not fit on a single line are wrapped over multiple rows. Scroll bars



Group: none | Alphabetical | ASC | Sort: Frequency | DESC | Glyph size: none

gap penalty: 2 | substitute penalty: 1

Size: | Layout: Vertical | Ignore traces: | Multi-match: ✓ | Stack: ✓

Context	#Packets	#Conversat
All	792518	180938
Invite only	169523	15347
Internal traffic	117463	10769

Selected traces: 2408(22.4%) Total: 10769 \| Selected messages: 22193(18.9%) Total: 15093

Fig. 4. Graphical user interface of the implemented prototype and components: A) Sequence view represents sequences as series of glyphs. Settings with respect to sorting, grouping, and clustering of sequences are set using controls in A) and G). B) The alignment view finds overlap in sequences of interest by aligning them. Alignment parameters, layout and sort settings can be modified in B) and H). C) The attribute view shows trends and patterns in selections on a per-attribute basis. D) Context view enables experts to store selections of interest throughout exploration. E) Rule overview widget to show the coverage of applied rules. F) Rule view shows a list of applied rules along with their settings. The ordering for rewriting is controlled via drag and drop operations. Rules can be toggled on or off along with longest and shortest match settings.

are used to inspect the entire data set. The size of the glyphs can be set proportional to a numeric attribute of choice. Since Eventpad focuses on the reduction and analysis of event patterns, time-intervals between events are disregarded in the visualization.

Users can replace one or more macroevents in sequences by a new one using a rule editor (Figure 3). Traditional regex operators such as sequential composition, choice, and iteration (0 or more times) can be used to construct patterns. Figure 2 shows how these operators are visually encoded in the interface. Similar to Word's equation editor [32], operators can be combined to construct more complex patterns.

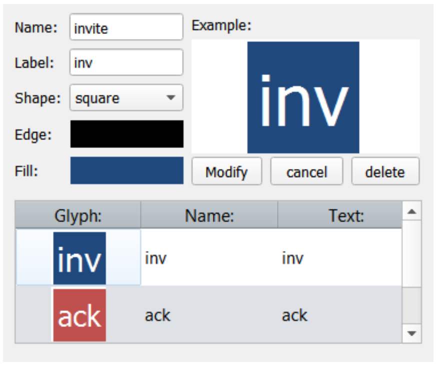
Users can specify a predicate by selecting a glyph of choice. The "Wilcard" glyph corresponds to any label. Double-clicking a glyph in the query interface results in a textual interface (Figure 3). This enables users to specify a predicate over attributes and values in the data. For the right hand side of a rule, users can design their own glyphs by choosing a particular shape, color and/or label (Figure 5). Earlier defined glyphs can be reused for the creation of new rules. In general, rules are used for three purposes (also depicted in Figure 3):

1. filtering,
2. highlighting, and
3. compression.

Filter rules are constructed by rewriting patterns to the empty macroevent and are typically used for data reduction. Highlight rules enable users to visually emphasize events of interest for their investigation, whereas compression rules group collections of events to reduce variety or repetition.

4.3 Rule interaction

In Eventpad, multiple rules can be applied one after each other. Rules are shown in a list (Figure 4F) and applied from top to bottom. The ordering of the rules can be rearranged using drag and drop operations and rules can be toggled on and off to study their effect. Regex rules can match patterns of varying length, e.g., application of the rule a.*b $\rightarrow$ R to a string abcb can lead to either Rcb or just R. Analysts can determine for every rule whether longest or shortest matching should be applied using the rule view.


Name: invite

Label: inv

Shape: square

Edge:

Fill:

Example:

Modify | cancel | delete

Glyph:	Name:	Text:
inv	inv	inv
ack	ack	ack

Fig. 5. Editor interface for the construction of custom glyphs.

A direct consequence of our extension to multivariate data is the possibility of having a macroevent adhere to multiple rules at the same time (also referred to as *multi-matching*). This is for instance the case when two rules specify constraints over two different attributes. In order to make users aware of this, in Eventpad's sequence view we visualize this overlap by stacking the corresponding glyphs of a macroevent on top of each other. An offset is used to ensure that previous matches are still visible to the user. Users can disable multi-matching to prevent glyphs from being stacked. In this situation, only the glyph of the last obtained macroevent label is shown to the user.

To study the impact of a rule over the data set, an icicle plot is introduced visualizing the fraction of glyphs that are rewritten by the current rule set (Figure 4E). The icicle plot is constructed in the same order as the rules are applied. The order in which the rules occur in the rule view determines the ordering in which the glyphs are stacked.

Besides rules, users can select patterns via a search interface. Users can stack glyphs in the search interface to enforce a macroevent to adhere to more than one rule at the same time. Inversely, the exact operator is used to ensure that macroevents of interest only adhere to the specified visual representation. Figure 6 shows a search dialog where both operators are applied.

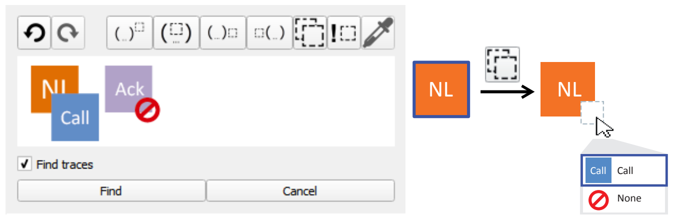


Fig. 6. Search interface where users are only interested in sequences where call events also originate from the Netherlands. In addition, these call events should be followed by ack events only (i.e., without overlap).

5 Pattern Aggregation

Although rules can significantly reduce the length and complexity of sequences, slight variations between sequences in real-world data are inevitable. In the next sections we describe how we use *clustering*, *alignment*, *sorting*, and *partitioning* operations to discover patterns between sequences through overview.

5.1 Structural sequence overlap

Depending on the type of rule it is possible for a glyph to represent more than one event. Users are informed about this by placing a red popup in the upper right corner of the glyph showing the number of events it contains. Since this popup does not alter the type of glyph, we decided to exclude this information during sorting and clustering.

Since all sequences typically do not fit on the screen, users can gain better insight in their event log by clustering sequences based on their glyph representation. This results in a stacked view only showing the unique sequences in the data set based on the currently defined set of rules. The frequency of a sequence is shown textually at the end of every sequence. Stacked sequences can be sorted by frequency to discover generic patterns or outliers. Users can also sort sequences according to various metrics:

- *Default* presents the sequences in the order of the input.
- *Alphabetical* sorts sequences by representing every sequence as a string. The string is obtained by concatenating the labels of glyphs together separated by delimiters.
- *Clustered* sorts sequences by applying single linkage agglomerative hierarchical clustering [13] on the unique event sequences based on Hamming distance.
- *Selected* sorts sequences by their number of currently selected events.

With hierarchical clustering sequences of similar length are positioned close to each other, whereas alphabetical sorts sequences based on their starting sequence. The Hamming distance $d(s_1, s_2)$ between two sequences s_1, s_2 is defined by their number of dissimilar glyphs:

$$\sum_{1 \leq i \leq min(\#s_1, \#s_2)} \left(\begin{cases} 0, & s_1[i] = s_2[i] \\ 1, & \text{otherwise} \end{cases} \right) + |\#s_1 - \#s_2|,$$

where $s_1[i]$ represents the i-th macroevent of s_1 and $\#s_1$ its corresponding length. Two macroevents are assumed to be equal if they are rewritten by the same set of rules.

To gain better insight in characteristics of particular sequences, users are enabled to partition sequences according to a sequence attribute of choice. This enables users for instance to inspect event sequences based on their length, duration, or starting time. By default all event sequences are contained in a group called “All” (Figure 5A).

5.2 Sequential sequence overlap

The wide variety in event sequences sometimes makes it difficult to detect potential overlap between them. We believe however that discovering overlap is key to understanding how deviations between sequences may have developed. The alignment view (Figure 4B) enables analysts to generate an overview visualization of event sequences of interest by aggregating them in an icicle plot.

Current alignment methods [44] often focus on the alignment of sequences by the n-th occurrence of an event. In Eventpad, we assist analysts in finding multiple areas of interest through Multiple Sequence Alignment (i.e., MSA). Figure 7 shows the effect of MSA on the traditional icicle plot. MSA is a popular technique in the area of bioinformatics to discover patterns between (fragments of) DNA sequences. As DNA sequences typically show overlap in small fragments of the sequences, they often use MSA algorithms with local optimization. Since we are interested in overlap and differences between entire sequences, we chose for an MSA algorithm focusing on global alignment.

Bose et al. showed the value of applying sequence alignment to gain better insight in event logs [6]. They also indicated the need for interaction to explore patterns in greater detail. In Eventpad analysts are enabled to apply the progressive global MSA algorithm by Bose et al. on event selections of interest to automatically find areas of overlap. Analysts can modify parameter settings such as gap cost to determine the amount of white spacing the MSA is allowed to introduce. Selecting a block in the alignment view results in the selection of its corresponding events in the sequence view. This enables users to select similar events across multiple sequences with a single click of a button.

6 Selections

Pattern aggregation enables users to perform high-level comparisons, but does not enable the inspection and comparison of multivariate data outside the scope of rules. In addition, analysts need to be enabled to focus on parts of the event sequences that are relevant for their investigations. For this we enable users to create selections of interest.

6.1 Context

The context view (Figure 4D) enables users to save selected events of interest into a new context by assigning a name to them. The creation of a new context results in a new attribute separating the selected events from the non-selected. This attribute is added to the data and can be used for further analysis and drill down, enabling analysts to tag the data with more domain-specific information throughout exploration. To stay aware of the impact of a particular selection, the status bar is used to display the number of events and sequences that are currently selected in the active context.

Similar to Cappers et al. [9] contexts are saved in a tree structure where the hierarchy shows the ordering in which the contexts are created. Context c is a child of parent context d if and only if c was created when the analyst was exploring d. Contexts are used to focus on smaller subsets of the data. Right clicking a context a while exploring b results in the selection of all events that a and b have in common.

6.2 Attributes

To enable the inspection of multivariate data in event selections of interest, an attribute view (Figure 4C) is introduced showing an overview of event attributes using scented widgets [45]. The histogram bins are interactive and can be used to select and deselect events with specific attribute values within the current context. Scented widgets for sequence attributes are introduced in a second tab. Selections can be enforced within specific value ranges of a widget by adjusting its scented span slider. For categorical attributes an exclamation mark is shown in front if the number of values is too large to visualize. To inspect selected values that are not visualized, histogram bins can be sorted by frequency, rarity, or whether they occur in the current selection. Since the number of attributes is typically larger than the number of widgets that fit on screen, we enable users to filter attributes by name using a textual interface.

6.3 Interaction

Event-oriented interaction is used to keep brushing and linking consistent and understandable over all the views. Users can modify selections of interest by selecting and deselecting visual elements across different views while holding the CTRL key. In case of the attribute view, blue bars are used to show the fraction of selected events in every bin. Glyphs whose event collections are partially selected are marked with a blue dashed border, solid otherwise. Double clicking a glyph in the

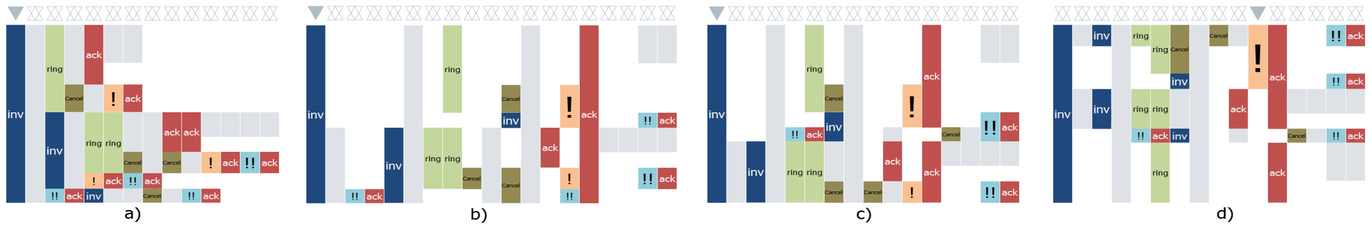


Fig. 7. Effect of alignment to event sequences: a) No alignment. b) MSA layout with gap cost = 1 c) MSA layout with gap cost = 2 d) Effect of sorting on different part of the alignment.

sequence view results in the selection of its corresponding sequence. Right clicking on a selection enables users to store selections for further investigation, invert selections, or inspect the multivariate properties of the selected events in a tabular view.

7 Use cases

We demonstrate the exploration method on two real-world multivariate event sequence data sets. We show how tight coupling between multivariate and sequential analysis is achieved using two use cases. In the first use case we start with the analysis of known sequential patterns in order to find anomalies and patterns in the multivariate data. In the second use case set we start with the analysis of multivariate data in order to discover sequential patterns of interest in the data set.

7.1 VoIP traffic

The proposed exploration method was designed in collaboration with a Dutch telecom company specialized in the provision of communication services over Internet using Voice over IP (VoIP).

7.1.1 Problem statement

For the establishment of a VoIP conversation handshaking signals such as invite (start call), acknowledge (accept call), cancel (interrupt call), and bye (end call) are transferred using a protocol called SIP [37]. Besides the type of signaling, these messages have additional information, such as status codes, source and destination phone numbers, (geo) domain information, user-agent etc. A conversation is uniquely defined by a `Call-Id`. The presence of illegal SIP sequences and/or invalid SIP messages in the traffic can cause SIP servers to go to an invalid state where conversations are no longer properly billed or secured, or could even lead to the disruption of the server [16].

A common attack model for hackers to abuse this state is to make money using Toll fraud [1]. In this model, hackers steal user credentials to hijack a company's VoIP phone, make many (long) phone calls to premium numbers they own in order to receive thousands of euros for the dialed numbers at the expense of the company. The detection of these Advanced Persistent Threats (APTs) [39] in general is difficult since APT and "normal" traffic look very similar.

The flexibility of the SIP protocol makes the distinction between valid and invalid phone conversations ill-defined. Depending on the vulnerabilities in new VoIP software updates, these definitions might even change over time. The main goal of the exploration is to find out whether their servers and customers properly send SIP messages conform the RFC standard [38]. Gaining insight in unexpected signaling can help analysts to understand whether they were caused due to bad server configurations or the presence of malicious users.

7.1.2 Exploration

Together with four analysts we organized an interactive session where we used Eventpad to study their SIP traffic. In their daily routine the analysts use applications such as VoiPMonitor [40] and the protocol-analyzer Wireshark [10] to gain better insight in their phone traffic.

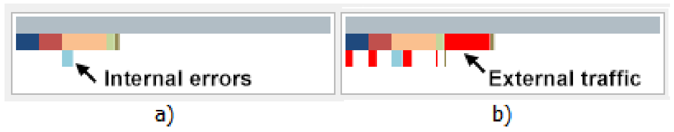


Fig. 8. a) Application of INVITE (blue), ACK (bordeaux), and error messages (orange/blue). b) Messages in red represent external traffic.

They often use search tools such as `grep` and Elasticsearch in their investigations to find explanations for errors in their server logs. We initially analyzed 800,000 SIP messages consisting of approximately 181,000 conversations and 60 attributes. The traffic was obtained by recording 20 minutes of SIP signaling from one of the data centers with a load of approximately 3000 concurrent calls per second.

We started exploration with a *black listing* approach where analysts created rules to search for known undesired SIP conversations. For this they created a rule where glyphs whose event's status code represent SIP client or server failure are replaced with an orange "!" glyph. Blue "!!" glyphs are introduced for error codes that were sent from their own servers. The rule overview showed that only 2% of the events contained internal errors (Figure 8a). They investigated this by sorting the sequences alphabetically and creating three rules to visually distinguish between `INVITE`, `ACK`, and `BYE` messages. The rule overview showed that only 35% of the data was covered by the new rules (Figure 8a). Inspecting the `sip.Method` widget showed that almost 60% of their traffic consisted of other SIP traffic involving `Option` messages to ping server information and `REGISTER` messages. In addition, they noticed a small percentage of `MESSAGE` events that are supposed to be deprecated in their platform because of known vulnerabilities [16].

To study the variety in the traffic, analysts decided to cluster the sequences and sort them by frequency. They were shocked to see that only 20% traffic was captured in the top 10 most frequent patterns (Figure 9B) as this indicates high variability and many deviations from expected standard behavior. Next, analysts filtered out incomplete conversations by searching for sequences starting with an invite request using the search interface. In addition, they decided to focus on their own traffic by excluding traffic from third party VoIP providers. This resulted in a new context and attribute called "Internal traffic".

After selecting the frequent patterns and aligning them using MSA, analysts saw that, despite the variety, overlap between sequences was high (Figure 4B). Analysts now noticed the presence of a proxy server in the middle of phone conversations (Figure 4B: two INV and two ACK messages nested). This made them realize that some phone calls were migrated to other data centers for load balancing. In cases where this proxy was not present, erroneous messages were generated twice in one sequence (highlighted in Figure 4B). In this sequence INV and ACK are not nested. Analysts selected the aligned "!!" events in the alignment view and inspected their multivariate data using a tabular view. Inspecting the `sip.From` header of the events revealed that most erroneous messages had an anonymous source phone number and an `invalid` domain (Figure 4C).

Based on previous observations, we organized a second session where we extended the analysis over a larger period in time. We excluded ping traffic, incomplete conversations due to fragmentation, and phone calls that were established outside their platform. In this session we incorporated two hours of traffic by simultaneously recording traffic from two data centers. Since Wireshark does not support any filtering mechanism at sequential level, we use the Eventpad engine for preprocessing the data set only considering conversations:

- that start with an `INVITE` message,
- where an invite should eventually be followed by a `BYE` or `CANCEL`, and
- only contains messages whose `From.host` and `To.host` are inside the company's domain.

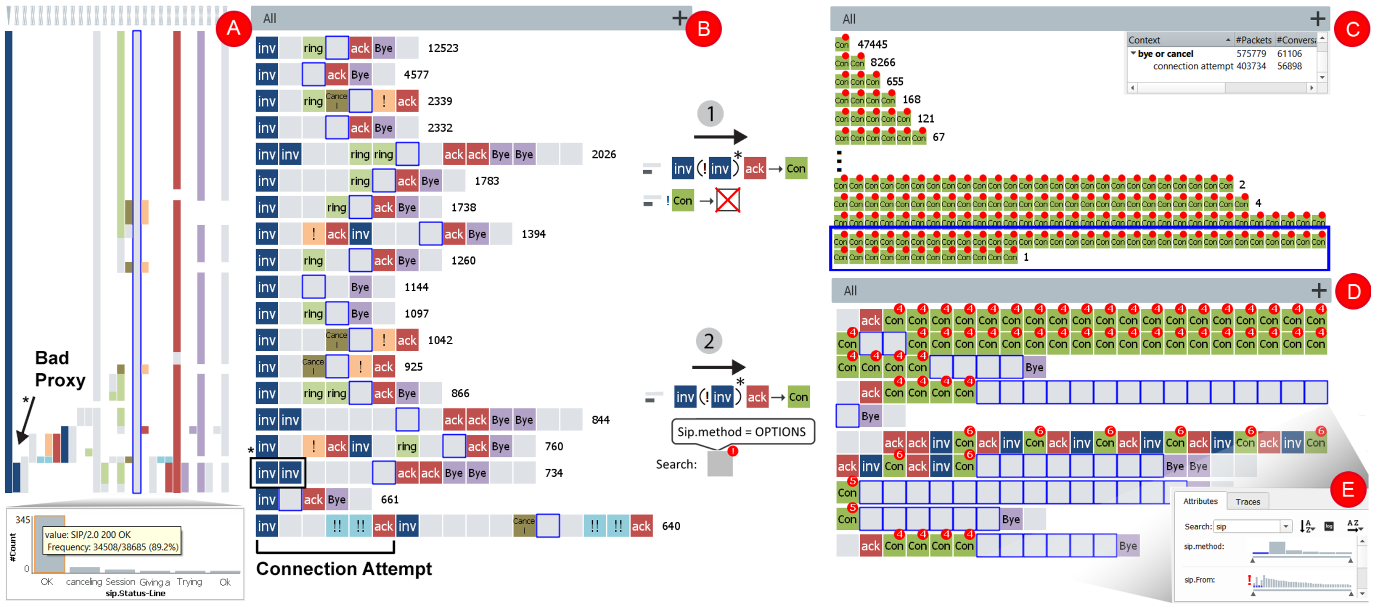


Fig. 9. A) Investigation of multivariate data of aligned undefined elements. B) Most frequent VoIP patterns after preprocessing the data. C) Detection of many connection attempts within one conversation. D) Presence of ping traffic in conversations. E) Traffic originating from the same source.

In order to avoid the presence of duplicate messages due to call redirection of proxies, phone calls are identified by their triple `ip.src`, `ip.dst`, and `Call-Id`. The application of these filter rules resulted in the reduction from 40,000,000 events and approximately 4,000,000 conversations to the analysis of 1,300,000 events and approximately 80,000 conversations respectively.

After applying the rules, analysts created a new alignment of the most frequent patterns to see that in this selection the variety in the traffic was significantly reduced (Figure 9A). The sequential occurrence of INVITE messages in the Dutch traffic showed the presence of a bad proxy configuration where the target computer and proxy were the same (Figure 9A, *).

Partitioning the traffic by `geoip.src` showed that these bad proxies were located in Dutch traffic only (Figure 10). Furthermore, analysts noticed that most international phone calls to the Netherlands failed at their first connection attempt. Since it is possible for phone conversation to have multiple connection attempts, analysts decided to extract these patterns by constructing a rule as specified in Figure 9B. Bad proxy settings were ignored using shortest match and by enforcing that invites are not inside the attempts. After extracting the attempts and sorting them alphabetically, analysts noticed that most conversations in their network required at most two connection attempts in order to succeed (Figure 9C). Some phone conversations however required over 40 attempts (Figure 9D). Although the start time and duration of the conversation did not show anything suspicious, inspection of the event attributes shows the presence of `OPTIONS` messages inside a regular phone call. Although such a sequence is valid with respect to the RFC, in practice this is highly uncommon. After selecting all conversations with `OPTIONS` messages and inspecting the `sip.From` attribute showed that all conversations originated from the same client (Figure 9E).

After the sessions, one of the analysts said “For us, the system is a business intelligence tool that can really help us in understanding what is actually happening in our platform.” Additional features such as integration with Wireshark and shortcut functionality to instantly remove selected patterns of interest was requested to speedup their analysis process.

7.2 Hospital records

To illustrate the effectiveness of the method in other domains, we also analyzed a real-world hospital data set provided by the BPIC11 contest [43] consisting of approximately 1000 sequences, 134,000 events, and approximately 130 attributes. Apart from anonymization, the hospital log stores for every patient of a Gynaecology department

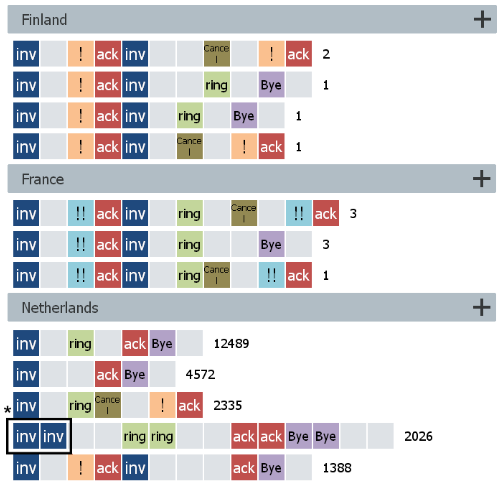
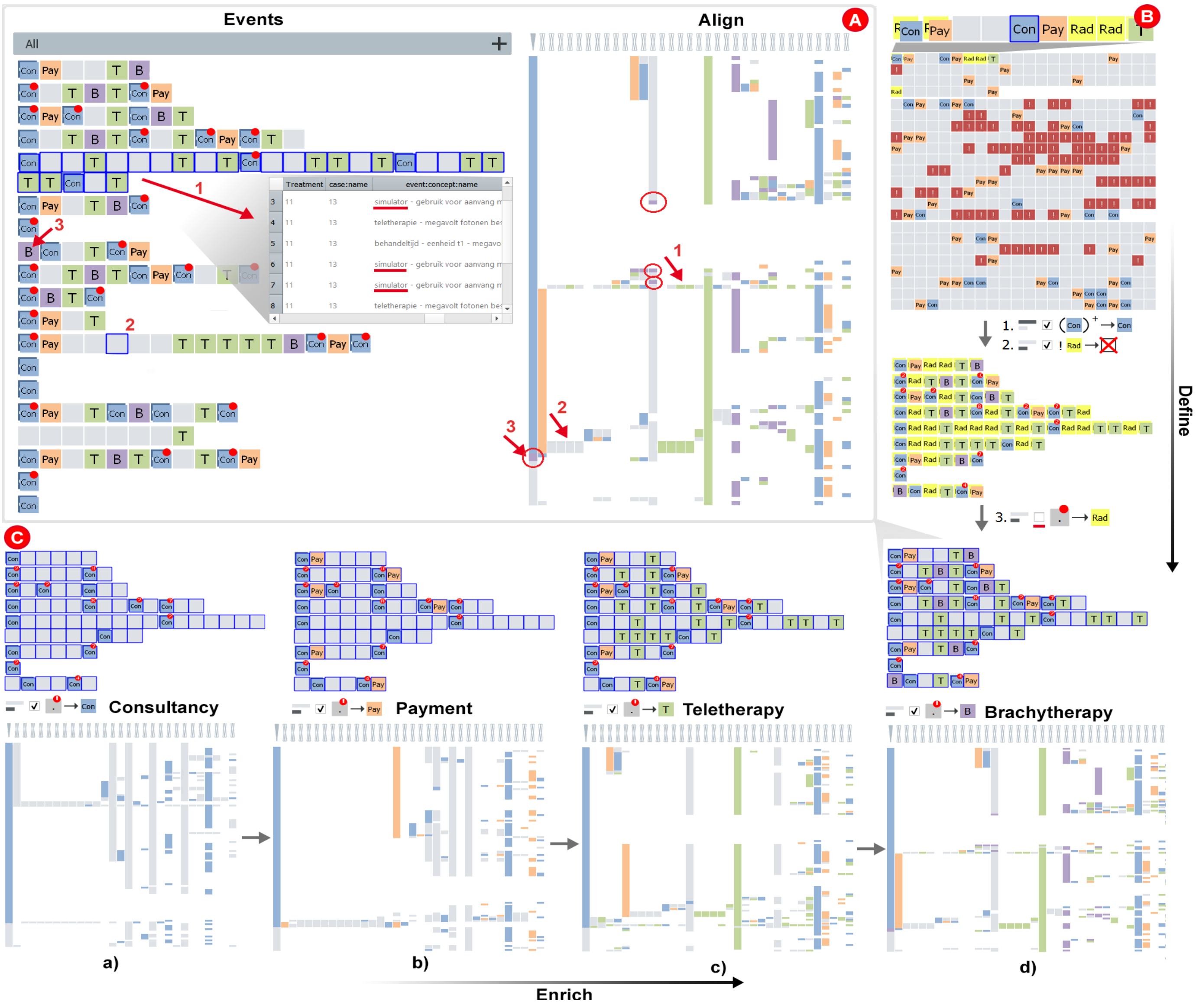


Fig. 10. Partitioning by `geoip.src` reveals that external calls contain errors and the presence of bad proxy configurations (*) in Dutch traffic.

when certain activities took place along with additional attributes such as which group performed the activity, the age of the patient, the activity’s level of emergency etc. The hospital log contains patient treatments where urgent activities are performed. In this use case we want to know when the hospital decides to make certain activities urgent. We try to find an explanation for these activities by testing whether urgent patient treatments share particular sequential patterns.

We start exploration by first creating a rule where all urgent events are marked in red (indicated in Figure 11B). We obtain all urgent sequences by searching for sequences with at least one red glyph and store them in a new context “urgent” (corresponding to 256 sequences). After selecting the new context, we inspect the `org_group` widget to see that all urgent events occur in the `General Lab`. To discover whether urgent events happened before or after certain treatments in other departments, we select events based on their `org_group` using the attribute view and inspect the number of cases that are involved via the status bar. This showed that in approximately half of the cases, the `Radiotherapy` group was involved in the treatment process.

Since radiotherapy is exceptional and only used for the treatment of cancer, we want to know whether the urgent activities happened before, after, or during this treatment. To explain the location of these events, we first have to understand the general workflow inside the department. We extract the radiotherapy events from the event log using the search



Treatment	case:name	event:concept:name
11	13	simulator - gebruik voor aanvang m
11	13	teletherapie - megavolt fotonen be
11	13	behandeltijd - eenheid t1 - megavo
11	13	simulator - gebruik voor aanvang m
11	13	simulator - gebruik voor aanvang m
11	13	teletherapie - megavolt fotonen be

Fig. 11. A) Investigation of anomalous alignments using selections and attribute inspection. B) Rewrite rules to study workflow patterns in the Radiotherapy department: 1. Group successive consultancy (blue) events in one glyph 2. Filter by radiotherapy (yellow) 3. Disable radiotherapy encoding. C) MSA results with gap cost 2 after incrementally adding rules to the data.

interface and store results in a new selection “radio”. We investigate the different activities using the `event:name` attribute widget. The number of different activities in the department is too large to visualize individually, but we can see that the most frequent activities can be grouped in four categories: consultancy (e.g., primary, secondary, and consultancy by phone), teletherapy (external radiation treatment), brachytherapy (internal radiation treatment), and payment administration. We define these categories by defining four rules where we search for events with keywords “consult”, “teletherapy”, “brachytherapy”, and “rate” in their activity name respectively. Since consultancy may require multiple sessions in a row that does not add value to the investigation, we define an additional rule where subsequent consultancy events are represented by a single blue glyph. Figure 11B shows the events of interest after applying the rules. Figure 11C shows the result of applying MSA to the glyph sequences before and after defining every rule one by one.

Applying MSA with gap cost of 2 shows that urgent radiotherapy treatments in general first start with consultancy (blue), basic treatment (gray), teletherapy (green) after which there is a concluding consultancy. Figure 12 shows the radiotherapy alignment along with the emergency activities indicated in red. Here we can see that in approximately half of the cases emergency events occurred before radiotherapy was started. We can also see that urgent events often occur between teletherapy (green) and brachytherapy (purple) sessions (Figure 12, black box). Inspection of these events shows that they analyze trombocytes and leukocytes in the patient’s blood to study the effect of the radiation. In non-urgent cases we discovered that this type of blood research is only performed during annual consultancies. This shows that the hospital assigns higher priority to blood results of patients during radiation treatment. Patients who still had urgent activities after brachytherapy were all diagnosed with either gynecological tumors or cervix uteri cancer (both highly uncommon in the data set).

Besides the general workflow, we discovered some anomalies that were unexpected:

- Some treatments involving brachytherapy did not receive consultancy in the Radiotherapy department (Figure 11A-3). Inspecting the sequences of interest outside Radiotherapy reveals that consultancy happened in a different department (Figure 11B).
- Some treatments show an increased number of undefined and teletherapy sessions after each other (Figure 11A-2). Closer inspection of the sequences in a tabular view reveals that treatments involve a rare activity named “simulator” (Figure 11A-1).
- In most cases brachytherapy is only performed after a teletherapy session (Figure 11A-3). However in 3% percent of the sequences

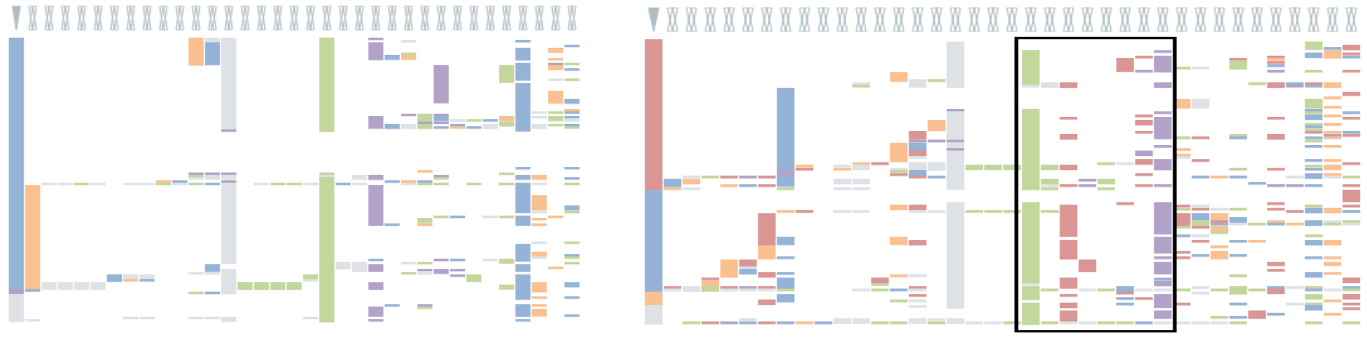


Fig. 12. General workflow radiotherapy department without (left) and with (right) urgent events in red. Urgent events between teletherapy (green) and brachytherapy (purple) mainly test for kalium, leukocytes and trombocytes in the patient's blood. Urgent events before and after therapy focus on calcium, glucose, natrium etc. Orange and blue events represent payment and consultancy respectively.

this did not hold (Figure 11, red circles). In these situations, all patients were diagnosed with `Malignancy endometrium` whose treatment apparently started all on the same day.

8 Discussion and Limitations

The use cases in Section 7 show how combined exploration of attributes and sequential analysis is achieved using rule-based event rewriting, pattern aggregation, and selections of interest as central elements. Each of these concepts has its own advantages and limitations for the analysis of events. We show that when combined they amplify each other. Tight linking between the concepts provides analysts a minimalistic yet expressive visual query mechanism to interactively select sequences of interest based on their sequential attributes, event patterns, and multivariate data associated with these events.

The definition of an anomalous sequence in general is ill-defined and requires domain knowledge and multiple iterations to define properly. The ability to visually encode parts of the data based on rules enables analysts to incrementally label their data and define their notion of what a good or bad sequence should look like. By sorting and clustering events based on the specified visual encodings analysts are able to study the coverage of their rules and discover new patterns of interest. Furthermore, the ability to evaluate rules in an offline setting also makes the tool suitable for data cleansing applications [29], where analysts can use the rules to only obtain the parts of the data (sequences) that are relevant for their investigation.

As with every technique, there are limitations. Although the application and construction of a rule in general is easy and intuitive to understand, the resulting rewriting can become complex when evaluating a large number of rules. During the interactive sessions with analysts we noticed that they needed around 10 active rules in order to answer their questions. The application of too many rules in parallel however can clutter the sequence and alignment view. Analysts can replace multiple rules by a new one by combining constraints over multiple attributes in one rule, but this only solves the problem partly. Although regular expressions in general are powerful to specify patterns, they have difficulties in capturing non deterministic behavior. Especially for the analysis of systems involving parallel communication, specification of patterns using regex can become cumbersome.

The effectiveness of clustering and alignment methods depends on the amount of information that is incorporated in the definition of the rules. In case of the hospital data we saw for instance that naively applying MSA without any proper rules resulted in no insights (underfitting) whereas creating rules for all possible (combinations of) values is tedious and results into noisy alignments (overfitting). Although it is possible for analysts to simplify sequences using rules and filtering techniques, domain knowledge about the underlying data is essential to obtain desired results. Especially when analyzing long sequences, coping with the variety in sequences and multivariate data can become difficult without a priori knowledge and expectations.

The approach suffers from a "cold-start problem" in the sense that users must already be aware of the information to be queried [31]. Although selections and scented widgets can help analysts in finding characteristics in parts of the data that are not captured by rules, this can be time-consuming without (automated) guidelines.

The performance of progressive MSA and single linkage hierarchical clustering in general are $\mathcal{O}(n^2m^2)$ and $\mathcal{O}(n^2m)$ respectively, where n represents the number of sequences and m the length of the sequences of interest. In practice we noticed that the application of the techniques to only unique patterns in the data set and selections of interest makes the analysis interactive for hundreds of sequences. However interactivity can be affected when naively applying these techniques over larger collections of long and unique sequences.

With respect to the visualization, the presented exploration approach focuses on the sequential occurrence of events rather than the time between them. In our problem statement absolute time does not play a major role. For applications where time between events is relevant for the analysis, the interface has to be adapted. Finally, if the number of attributes in events and sequences is large, interaction with attributes is limited to the number of visible scented widgets. Although sorting, filtering, and scrolling helps at finding attributes of interest, specifying queries involving many attributes becomes time-consuming.

9 Conclusion and Future Work

We presented a novel approach for analysts to explore multivariate event sequence data by combining attribute and sequential analysis into one unified system. The ability to interactively encode event logs by coalescing event sequences according to rules enables analysts to incorporate their knowledge to the data and test whether this matches their expectations. The combination of attribute-based scented widgets and pattern aggregations enables analysts to discover new attributes of interest and refine rules based on their new findings while staying aware of high-level patterns across different levels of abstractions. We have shown the effectiveness of the approach on real-world data sets through interactive sessions with external companies and elaborate examples. Since the methodology makes no underlying assumptions on sequential data, it is general and flexible enough to be used in other domains.

For future work it is interesting to see how we can extend the visualization paradigm to support more complex regex operators such as *capture groups* and *back referencing* [15]. Also, an extension of the glyph design interface to enable further parametrization of glyphs can help to study correlations between two or more attributes.

The system currently focuses on defining new attributes of interest at the level of an event. Applications where multivariate data is difficult to obtain (e.g., flow-based network traffic analysis [27]) however often focus on sequential properties in their analysis. Defining proper interaction schemes to support the creation of attributes at sequential level however is nontrivial, since they apply to a different level of abstraction.

Acknowledgments

This work is funded by SpySpot, a project in the Cyber Security program of Netherlands Organisation for Scientific Research (NWO). The authors would like to thank Kay Reijnen and Jeroen Teeuwen et al. from Motto Communications, and in particular SpySpot members Sandro Etalle, Ömer Yüksel, Davide Fauri, and Jerry den Hartog for their valuable comments throughout this work.

References

[1] H. Abdelnur, T. Avanesov, M. Rusinowitch, and R. State. Abusing sip authentication. In *Information Assurance and Security, 2008. ISIAS'08. Fourth International Conference on*, pp. 237–242. IEEE, 2008.

[2] A. Aho, R. Sethi, and J. Ullman. *Compilers, Principles, Techniques*. Addison wesley Boston, 1986.

[3] J. Avallone. regexper. https://regexper.com/.

[4] J. Bernard, D. Sessler, T. May, T. Schlomm, D. Pehrke, and J. Kohlhammer. A visual-interactive system for prostate cancer cohort analysis. *IEEE computer graphics and applications*, 35(3):44–55, 2015.

[5] J. Bernard, M. Zeppelzauer, M. Sedlmair, and W. Aigner. A Unified Process for Visual-Interactive Labeling. *EuroVis Workshop on Visual Analytics*, 2017.

[6] R. Bose and W. van der Aalst. Trace alignment in process mining: opportunities for process diagnostics. In *International Conference on Business Process Management*, pp. 227–242. Springer, 2010.

[7] M. Brehmer and T. Munzner. A multi-level typology of abstract visualization tasks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2376–2385, 2013.

[8] Y. Cai and R. de M. Franco. Interactive visualization of network anomalous events. In *International Conference on Computational Science*, pp. 450–459. Springer, 2009.

[9] B. Cappers and J. van Wijk. Understanding the Context of Network Traffic Alerts. In *Visualization for Cyber Security (VizSec), 2016 IEEE Symposium on*, pp. 1–8. IEEE, 2016.

[10] G. Combs et al. Wireshark-network protocol analyzer: http://www.wireshark.org/. *Version 0.99*, 5, 2008.

[11] F. Du, B. Shneiderman, C. Plaisant, S. Malik, and A. Perer. Coping with volume and variety in temporal event sequences: Strategies for sharpening analytic focus. *IEEE transactions on visualization and computer graphics*, 2016.

[12] S. Eick, M. Nelson, and J. Schmidt. Graphical analysis of computer log files. *Communications of the ACM*, 37(12):50–56, 1994.

[13] B. Everitt, S. Landau, M. Leese, and D. Stahl. Hierarchical clustering. *Cluster Analysis, 5th Edition*, pp. 71–110, 2011.

[14] J. Fails, A. Karlson, L. Shahamat, and B. Shneiderman. A visual interface for multivariate temporal data: Finding patterns of events across multiple histories. In *Visual Analytics Science And Technology, 2006 IEEE Symposium On*, pp. 167–174. IEEE, 2006.

[15] J. E. Friedl. *Mastering regular expressions*. " O'Reilly Media, Inc.", 2002.

[16] D. Geneiatakis, T. Dagiuklas, G. Kambourakis, C. Lambrinoudakis, S. Gritzalis, S. Ehlert, D. Sisalem, et al. Survey of security vulnerabilities in session initiation protocol. *IEEE Communications Surveys and Tutorials*, 8(1-4):68–81, 2006.

[17] C. Gormley and Z. Tong. *Elasticsearch: The Definitive Guide*. " O'Reilly Media, Inc.", 2015.

[18] D. Gotz and H. Stavropoulos. Decisionflow: Visual analytics for high-dimensional temporal event sequence data. *IEEE transactions on visualization and computer graphics*, 20(12):1783–1792, 2014.

[19] A. Jain, N. Murty, and P. Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[20] B. W. Kernighan and R. Pike. *The Unix programming environment*, vol. 270. Prentice-Hall Englewood Cliffs, NJ, 1984.

[21] S. Kleene. Representation of events in nerve nets and finite automata. Technical report, DTIC Document, 1951.

[22] H. Koike and K. Ohno. Snortview: Visualization system of snort logs. In *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, VizSEC/DMSEC '04, pp. 143–147. ACM, New York, NY, USA, 2004. doi: 10.1145/1029208.1029232

[23] J. Krause, A. Perer, and H. Stavropoulos. Supporting iterative cohort construction with visual temporal queries. *IEEE transactions on visualization and computer graphics*, 22(1):91–100, 2016.

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

[25] T. Kunz. Visualizing abstract events. In *Proceedings of the 1994 Conference of the Centre for Advanced Studies on Collaborative Research*, CASCON '94, pp. 38–. IBM Press, 1994.

[26] 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*, pp. 147–154. IEEE, 2007.

[27] B. Li, J. Springer, G. Bebis, and M. H. M.H. Gunes. A survey of network flow applications. *Journal of Network and Computer Applications*, 36(2):567–581, 2013.

[28] Z. Liu, Y. Wang, M. Dontcheva, M. Hoffman, S. Walker, and A. Wilson. Patterns and sequences: Interactive exploration of clickstreams to understand common visitor paths. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):321–330, 2017.

[29] J. I. Maletic and A. Marcus. Data Cleansing: Beyond Integrity Analysis. In *Iq*, pp. 200–209. Citeseer, 2000.

[30] 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, pp. 38–49. ACM, New York, NY, USA, 2015. doi: 10.1145/2678025.2701407

[31] G. Marchionini. Exploratory search: from finding to understanding. *Communications of the ACM*, 49(4):41–46, 2006.

[32] Microsoft. Word equation editor (2010). support.office.com/en-US/article/Write-insert-or-change-an-equation-1D01CABC-CEB1-458D-BC70-7F9737722702/.

[33] M. Monroe. *Interactive event sequence query and transformation*. PhD thesis, University of Maryland, 2014.

[34] M. Monroe, R. Lan, H. Lee, C. Plaisant, and B. Shneiderman. Temporal event sequence simplification. *IEEE transactions on visualization and computer graphics*, 19(12):2227–2236, 2013.

[35] M. Monroe, K. Wongsuphasawat, C. Plaisant, B. Shneiderman, J. Millstein, and S. Gold. Exploring point and interval event patterns: Display methods and interactive visual query. *University of Maryland Technical Report*, 2012.

[36] J. Pearlman and P. Rheingans. Visualizing network security events using compound glyphs from a service-oriented perspective. In *VizSEC 2007*, pp. 131–146. Springer, 2008.

[37] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. Sip: session initiation protocol. Technical report, MIT, Columbia University, 2002.

[38] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, and E. Schooler. Session initiation protocol (sip) rfc 3261. *IETF, A*, 2002.

[39] R. Sloan. Advanced Persistent Threat. *Engineering & Technology Reference*, 1(1), 2014.

[40] Solarwinds. Voipmonitor. http://www.voipmonitor.org/.

[41] C. Tominski. Event-Based Concepts for User-Driven Visualization. *Information Visualization*, 10(1):65–81, 2011. doi: 10.1057/ivs.2009.32

[42] J. Turnbull. *The Logstash Book*. James Turnbull, 2013.

[43] B. van Dongen. Business processing intelligence challenge (BPIC). http://www.win.tue.nl/bpi/doku.php?id=2011:challenge, 2011.

[44] T. Wang, C. Plaisant, A. Quinn, R. Stanchak, S. Murphy, and B. Shneiderman. Aligning temporal data by sentinel events: discovering patterns in electronic health records. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 457–466. ACM, 2008.

[45] W. Willett, J. Heer, and M. Agrawala. Scented widgets: Improving navigation cues with embedded visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1129–1136, 2007.

[46] K. Wongsuphasawat. *Interactive exploration of temporal event sequences*. PhD thesis, University of Maryland, 2012.

[47] K. Wongsuphasawat, G. G. Gómez, C. Plaisant, T. 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*, pp. 1747–1756. ACM, 2011.

[48] K. Wongsuphasawat and D. Gotz. Exploring flow, factors, and outcomes of temporal event sequences with the outflow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2659–2668, 2012.

[49] E. Zgraggen, S. Drucker, D. Fisher, and R. Deline. (s—qu) eries: Visual Regular Expressions for Querying and Exploring Event Sequences. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pp. 2683–2692. ACM, 2015. doi: 10.1145/2702123.2702262

[50] 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*, pp. 259–268. ACM, 2015.