EuroVis Workshop on Visual Analytics (2015)
E. Bertini and J. C. Roberts (Editors)

Exploration and Assessment of Event Data

Peter Bodesinsky , Bilal Alsallakh , Theresia Gschwandtner and Silvia Miksch

Vienna University of Technology, Vienna, Austria

Abstract

Event data is generated in many domains, like business process management, industry or healthcare. These datasets are often unstructured, exhibit variant behaviour, and may contain errors. Before applying automated analysis methods, such as process mining algorithms, the analyst needs to understand the dependency between events in order to decide which analysis method might fit the recorded events. We define a categorization scheme of event dependencies and describe a preliminary approach for exploring event data, combining visual exploration with pattern mining. Events of interest can be selected, grouped, and visually explored, using either a sequential or a temporal scale. We present two use cases with shopping event data and report expert feedback on our approach.

Categories and Subject Descriptors (according to ACM CCS): H.2.8 [Database Management]: Database Application—Data Mining H.5.2 [Information Interfaces and Presentation]: User Interfaces—

1. Introduction

Analysing and understanding event data is essential to optimize business processes. Events are associated with an event source, or a case, which generates an event sequence. Order and execution time of events are defined by timestamps. Various techniques for mining event data are available, most commonly, frequent sequential pattern [AS95] and association rule mining [AS94]. Process Mining [vdA*12, vdA11] deals with mining event logs to discover, check and enhance processes. Before applying automated algorithms complex data has to be made understandable to find recurring patterns and subsequences in event data, group the data and check for errors [BMvdA13]. Thus, we propose a Visual Analytics (VA) approach which aims at supporting analysts in the initial exploration and assessment of event data. Our approach, described in Sect. 3, provides means for (1) browsing individual sequences using both sequential and temporal scaling (2) an overview of event and pattern frequency (3) perform pattern mining and inspecting the location of (recurring) patterns within event sequences. We define a categorization of event dependencies that commonly exist in event data and are crucial to identify in order to apply an appropriate mining method. Our prototype is focused on the exploration of the sequential dependency level, as a first step towards an integrated approach. In Sect. 4 we demonstrate the approach with two use cases for mining event-based shopping data and discuss user feedback in Sect. 5.

2. Related Work

Techniques for the analysis and visualization of event-based data are widespread. Workflow and process models derived from event data are often represented in a graph or flowchart-like manner [vdAWM04, GvDA07]. Techniques for event data aggregation that are based on Sankey diagrams [RHF05], are, for example, Outflow [WG12] and Frequency [PW14]. Other approaches, which mine and display frequent event sequences, are ActiviTree [VJC09], VizTree [LKL05], EventFlow [MLL*13] and the approach proposed by Wei et al. [WSSM12]. MatrixWave [ZLD*15] uses multiple transition matrices to show the flow of events. Cloudlines [KBK11] shows aggregated event counts in a time-oriented view.

Other approaches show individual event sequences. Dotted charts [SvdA07] show the distribution of events over time. Bose and van der Aalst [BvdA10] apply methods from biology for event sequence alignment. Event Tunnel [SOSG08] applies the metaphor of a 3D cylinder together with different arrangement strategies. Arc diagrams [Wat02] show repeating patterns in sequences. Approaches for the visualization of sequences have also been proposed in biology, like for comparing or aggregating DNA sequences [ADG11]. Existing approaches are focused on pattern visualization, or on showing individual events. In contrast we combine a pattern overview and an event view, showing individual events, in an interactive exploration environment.

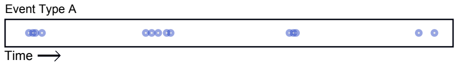
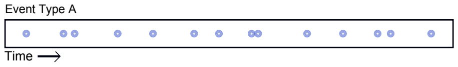
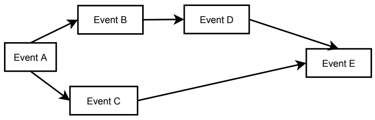
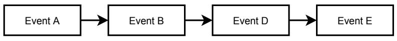
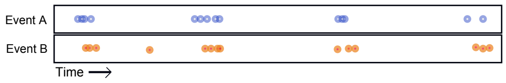
Dependency Level	Description	Example	Support
Single Event Type			
Time dependent	The occurence of an event type can be (at least coarsly) modeled. Its occurence depends on time.		Partial
No Dependency	The occurence of an event type is more or less random.		Partial
Multiple Event Types			
Process	Events appear in structured sequences. The succession of events can be desribed by rules defined in a workflow (i.e. process models).		Partial
Subset Process	A part of events appear in structured sequences and can be desribed by rules.		Partial
Strict Sequential Dependeny	A subset of events tend to occur in strict sequence together, but can not be described by a workflow.		Full
Loose Dependency	Some events exhibit dependency (but not necessarly follow or precede each other directly) or tend to occur together.		None
Loose Occasional Dependency	Some events exhibit dependency or tend to occur together, at least for some time spans.		None
No Dependency	Events occur independent from each other.		None

Table 1: Event dependency categorization scheme, **along with support level in our prototype**. The dependency levels are divided depending on the multiplicity of event types and sorted from highest to lowest within each group.

3. Approach

We present our categorization of event dependency in Table 1. Events are defined to be a realization of their specific event type. Occurrences of certain events can be time dependent or completely random. Realizations of multiple event types which exhibit strict dependency occur in structured sequences. They can be described by workflow rules (*process, subset process*) or by a set of common sequences (*strict sequential*). Realizations of multiple event types show loose dependency, if they tend to occur together, or if they follow/precede with varying time or other events in between (*loose, loose occasional*).

Our research prototype aims to help the user to determine the characteristics of event-based data and is focused on the exploration of strict sequential dependency (see Table 1). Our primary goal is to provide a versatile tool to assess event data. At first the user browses the dataset to gain an overview of event and pattern frequency and to develop initial hypothesis about their dependencies. Custom event grouping, coloring and labeling for events of interest can be performed and certain dependency patterns in these events can be examined in detail.

3.1. Event View and Pattern Representation

We encode each event of a case as a rectangular bar whose color represents the event type (see Fig. 1). All events of a case are aligned horizontally as a sequence of bars. Cases are arranged vertically beneath each other. Our example dataset (see Sect. 4.1) contains the recorded requests of a webservice connected to a webshop. Each user session (i.e. one user browsing the shop) represents a case. The type of the event can either be a *request for related webshop items* (R), a *buy request* (B), or a *view request* for an item (V). Repeating occurrences of a selected pattern inside a case are connected by semicircular arcs, similar to the arc diagram visualization [23]. The semi circles facilitate following repeating patterns and gaining insight into their distribution. If a pattern occurs only once within a case, it is marked by a rectangle.

The whole dataset and all cases can be browsed rapidly by scrolling in the same way as for documents or webpages. We support different ordering modes of cases. *Sort by time*, as a default mode, orders cases by the timestamp of the first event. *Sort by frequency* of a selected pattern X (see Sect. 3.2) orders the cases according to the number of occurrences of X within their events. *Sort by sequence length* orders the cases by their number of events.

Our prototype enables the user to assign color and labels to event types of interest. Coloring certain event types emphasises them and allows to analyse how they are related to each other. Furthermore our approach allows to group multiple event types of low granularity. Grouped event types are assigned the same color and treated as one abstract event in pattern mining, as explained next.

3.2. Pattern Overview and Interactive Mining

We combine pattern overview and mining with the event view. We perform simple pattern matching based on a sliding window and count for each occurrence of the same se-

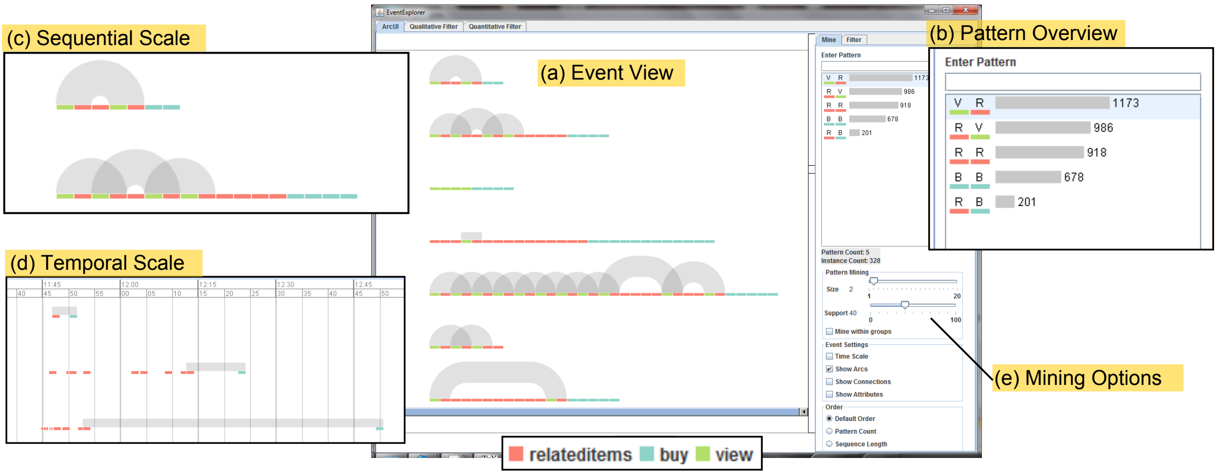


Figure 1: Interface. (a) Events of a case are shown as a horizontal sequence of bars. Cases (event sources) are arranged vertically. Recurring instances of the selected pattern are connected by arcs. (b) A vertical bar chart shows pattern frequency. (e) Pattern size and support can be defined. (c) Sequential scaling. (d) Temporal scaling emphasizes the actual time of events.

quence. A pattern can be defined either interactively by selecting an event sequence of interest within the event view or by entering the sequence as a regular expression. Furthermore our approach supports automatic mining of frequent patterns, with user-defined pattern size and support threshold (see Fig. 1e).

Mined as well as manually-entered patterns are shown as a bar chart (see Fig. 1b). Bar length represents the pattern frequency (i.e. how often the pattern occurs in the log). The pattern itself is shown as a sequence of colored rectangular bars, just like in the event view. A selected pattern is highlighted in the event view using arcs. Mining for patterns of size one shows the frequency of each event type in the log. Mining patterns of size two (two events in sequence) gives clues about possible casual relationships between events. Findings about event causality are the basis for generating hypothesis and for constructing process models. Patterns of a larger size help to identify repetitive behaviour, possible loops, or subprocesses. Inspecting events and patterns allows to estimate if a process model might fit to explain the observations, thus we partially support (subset) process dependency (see Table 1). A video showing interactive mining is available in the supplementary material of the paper.

3.3. Scaling Options and Filtering

Different scaling policies are supported. Sequential scaling (see Fig. 1c) renders succeeding events in an equally spaced grid, no matter how much time passed between them. Partial support for time dependency (see Table 1) is provided by temporal scaling (see Fig. 1d). It allows to see how much time has passed between the events. Switching the scaling helps to reveal casual as well as temporal features of the dataset. Moreover, we provide a filter functionality to filter the cases with respect to event attributes. It can be used to select a subset of cases which exhibit a specific behaviour or are of specific event types.

4. Use Cases

We apply our approach to event-based shopping data, which contains information on past purchases and transactions. Customer data is often used for product recommendations and to analyze customer response to certain offers. In contrast with other data sources related to these tasks, such as shopping baskets and customer preferences, event-based data reveals dynamic purchase behaviour. Initial exploration of such data sources is crucial for the analysts to determine their value and to decide on the mining method. Another goal is to identify interesting behaviour and patterns that might help to explain, for instance, how different purchases are related to each other.

4.1. Webservice Log

The first use case is the exploration of a webservice log connected to a webshop (see Sect. 3.1). As shown in Fig. 1a and 1c *view requests (V)* are often followed by *requests for related items (R)* (pattern VR is selected and highlighted). *Buy requests (B)* rather happen in the end of a session, the majority of users tend to explore the shop (*view requests* and *related requests*) before they make their purchase. To further examine this hypothesis the analyst can mine for patterns of size two and display the pattern counts in the pattern list (Fig. 1b). This reveals that *buy requests* preceded by *related item requests* (RB) is a common pattern. Selecting this pattern in the list displays this pattern in the event view, which confirms that one or multiple *buy requests* are preceded by a longer search for items. Switching to the temporal scale gives insight about session duration (see Fig.1d). Most sessions do not take more than one hour. *Buy requests* tend to happen with a little delay after bursts of events related to browsing the shop. Furthermore event abstraction might be used to merge related item and view requests into one event named "browsing" to visually emphasize this behaviour, i.e. that buy requests happen in the end.

4.2. Transaction Data

Transactions contain information about payment events of a customer. Each recorded transaction event consists of different fields, like a timestamp, the purchase amount, the merchant type and the Id of the customer. An example would be a transaction made on 17.11.2014 at 16:00, with a purchase amount of 50 euro at a music store by customer 540503. Transactions of a customer are considered to be a case (customer is the event source) and the merchant type is considered to be the event type. Initial analysis suggests that the sequences of the dataset are rather unstructured. Filtering is first applied by the analyst to reduce the working set to interesting cases. For example, to analyse the behaviour of travelling customers, we can include all cases in the set which contain at least one transaction associated with a hotel payment. In the next step custom colors and labels can be assigned to certain event types, such as hotel (H), airline (A) and restaurants (R) payments. Mining for frequent event types gives insight that travelling customers seem to purchase at supermarkets and also use computer network services. Switching to the temporal scale reveals that some events types occur periodically, for instance, on a daily basis (like restaurant visits).

5. User Feedback

We conducted an informal user feedback session with four users, which work for a company that deals with analysing and mining customer data. Our goal was to gather qualitative feedback to find out if the users understand the design and if functionality needs extension or refinement. After a short introduction the users were asked to solve a set of tasks. Finally they freely explored the prototype and expressed their opinion. The prototype and the visual design was generally appreciated by all the users. Some issues were related to usability, for instance, most users expected immediate response to the interaction with the sliders for pattern size and support and did not realize that they had to hit a button first. Grouping, abstraction, and labelling of events was also suggested by the users, during as well as after exploring the prototype. Other suggestions were to show the actual time of events and to allow filtering of cases. All these suggestions are already integrated in the current design. Scalability to large datasets having a large number of event types was also discussed, and is still an open issue (see Sect. 6). We plan to conduct further feedback sessions, as well as to use additional event datasets in the future to refine and extend our approach.

6. Discussion and Future Work

Our approach is still preliminary and has some limitations. An important issue to address is scalability. The limited number of well-distinguishable colors limits the types of events that can be emphasized simultaneously. Another challenge is to display multiple patterns simultaneously without cluttering. Appropriate scaling and aggregation mechanisms as well as a pixel-based visualization need to be investigated to gain an overview of a large number of cases and events, without the need for scrolling. Ordering the cases by similarity can help to find and group cases with similar behaviour. Different analytical and visual means need to be investigated to support analysis of event data with a loose level of dependency in the future (Table 1). For example, a view which shows the aggregated frequency of event types over time can reveal correlation between different event types as in Cloudlines [KBK11]. Fuzzy pattern mining algorithms are needed to detect specific event patterns despite variations and missing values in event sequences. Our future work is to support the assessment of event dependencies in a given dataset and to give experts a versatile tool to quickly analyse event-based data.

7. Conclusion

Analyzing event data is gaining importance due to the growing volumes of event data being recorded. It is necessary to inspect and assess event logs before applying mining algorithms. The degree of dependency between events dictates the appropriate automated methods to apply. We define a categorization for event dependency in event-based data and propose a Visual Analytics approach to explore and analyse dependency patterns in the data. Our approach supports exploring sequential dependencies by visualizing event sequences as well as the results of pattern mining algorithms. Furthermore, abstraction, filtering, and other interactions allow detailed analysis of certain patterns to develop hypotheses about the data. Two use cases for mining shopping event data demonstrate the applicability of our approach and the insights it can provide in event data. Our approach is still preliminary and restricted to small data sets having few hundreds of event sequences and a handful of event types. By integrating more advanced visualizations and pattern mining algorithms, our approach can be extended to provide overview of larger datasets and to reveal more complex dependency patterns in the data.

Acknowledgement This work was supported by Centre for Visual Analytics Science and Technology CVAST (funded by Austrian Federal Ministry of Science, Research, and Economy in the exceptional Laura Bassi Centres of Excellence initiative, project number: 822746)).

References

[ADG11] Albers D., Dewey C., Gleicher M.: Sequence surveyor: Leveraging overview for scalable genomic alignment visualization. *Visualization and Computer Graphics, IEEE Transactions on 17*, 12 (Dec 2011), 2392–2401. doi:10.1109/TVCG.2011.232. 1

[AS94] Agrawal R., Srikant R.: Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases* (San Francisco, CA, USA, 1994), VLDB '94, Morgan Kaufmann Publishers Inc., pp. 487–499. URL: http://dl.acm.org/citation.cfm?id=645920.672836. 1

[AS95] Agrawal R., Srikant R.: Mining sequential patterns. In *Data Engineering, 1995. Proceedings of the Eleventh International Conference on* (1995), IEEE, pp. 3–14. 1

[BMvdA13] Bose R., Mans R., van der Aalst W.: Wanna improve process mining results? In *Computational Intelligence and Data Mining (CIDM), 2013 IEEE Symposium on* (2013), pp. 127–134. doi:10.1109/CIDM.2013.6597227. 1

[BvdA10] Bose R., van der Aalst W.: Trace alignment in process mining: Opportunities for process diagnostics. In *Proceedings of the 8th International Conference on Business Process Management* (Berlin, Heidelberg, 2010), BPM'10, Springer-Verlag, pp. 227–242. URL: http://dl.acm.org/citation.cfm?id=1882061.1882084. 1

[GvDA07] Günther C., van Der Aalst W.: Fuzzy mining: Adaptive process simplification based on multi-perspective metrics. In *Proceedings of the 5th International Conference on Business Process Management* (Berlin, Heidelberg, 2007), BPM'07, Springer-Verlag, pp. 328–343. URL: http://dl.acm.org/citation.cfm?id=1793114.1793145. 1

[KBK11] Krstajic M., Bertini E., Keim D.: Cloudlines: Compact display of event episodes in multiple time-series. *Visualization and Computer Graphics, IEEE Transactions on 17*, 12 (Dec 2011), 2432–2439. doi:10.1109/TVCG.2011.179. 1, 4

[LKL05] Lin J., Keogh E., Lonardi S.: Visualizing and discovering non-trivial patterns in large time series databases. *Information Visualization 4*, 2 (2005), 61–82. doi:10.1057/palgrave.ivs.9500089. 1

[MLL*13] Monroe M., Lan R., Lee H., Plaisant C., Shneiderman B.: Temporal event sequence simplification. *Visualization and Computer Graphics, IEEE Transactions on 19*, 12 (Dec 2013), 2227–2236. doi:10.1109/TVCG.2013.200. 1

[PW14] Pererand A., Wang F.: Frequence: Interactive mining and visualization of temporal frequent event sequences. In *Proceedings of the 19th International Conference on Intelligent User Interfaces* (New York, NY, USA, 2014), IUI '14, ACM, pp. 153–162. URL: http://doi.acm.org/10.1145/2557500.2557508, doi:10.1145/2557500.2557508. 1

[RHF05] Riehmann P., Hanfler M., Froehlich B.: Interactive sankey diagrams. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on* (Oct 2005), pp. 233–240. doi:10.1109/INFVIS.2005.1532152. 1

[SOSG08] Suntinger M., Obweger H., Schiefer J., Gröller M.: Event tunnel: Exploring event-driven business processes. *Computer Graphics and Applications, IEEE 28*, 5 (2008), 46–55. doi:10.1109/MCG.2008.97. 1

[SvdA07] Song M., van der Aalst W.: Supporting process mining by showing events at a glance. In *Proceedings of the 17th Annual Workshop on Information Technologies and Systems (WITS)* (2007), pp. 139–145. 1

[vdA11] van der Aalst W.: *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, 1st ed. Springer Publishing Company, Incorporated, 2011. doi:10.1007/978-3-642-19345-3. 1

[vdA*12] van der Aalst W., et al.: Process mining manifesto. In *Business Process Management Workshops*, Daniel F., Barkaoui K., Dustdar S., (Eds.), vol. 99 of *Lecture Notes in Business Information Processing*. Springer Berlin Heidelberg, 2012, pp. 169–194. URL: http://dx.doi.org/10.1007/978-3-642-28108-2_19, doi:10.1007/978-3-642-28108-2_19. 1

[vdAWM04] van der Aalst W., Weijters T., Maruster L.: Workflow mining: discovering process models from event logs. *Knowledge and Data Engineering, IEEE Transactions on 16*, 9 (2004), 1128–1142. doi:10.1109/TKDE.2004.47. 1

[VJC09] Vrotsou K., Johansson J., Cooper M.: Activitree: Interactive visual exploration of sequences in event-based data using graph similarity. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (Nov. 2009), 945–952. URL: http://dx.doi.org/10.1109/TVCG.2009.117, doi:10.1109/TVCG.2009.117. 1

[Wat02] Wattenberg M.: Arc diagrams: Visualizing structure in strings. In *Information Visualization, 2002. INFOVIS 2002. IEEE Symposium on* (2002), IEEE, pp. 110–116. doi:10.1109/INFVIS.2002.1173155. 1

[WG12] Wongsuphasawat K., Gotz D.: Exploring flow, factors, and outcomes of temporal event sequences with the outflow visualization. *Visualization and Computer Graphics, IEEE Transactions on 18*, 12 (Dec 2012), 2659–2668. doi:10.1109/TVCG.2012.225. 1

[WSSM12] Wei J., Shen Z., Sundaresan N., Ma K.-L.: Visual cluster exploration of web clickstream data. In *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on* (Oct 2012), pp. 3–12. doi:10.1109/VAST.2012.6400494. 1

[ZLD*15] Zhao J., Liu Z., Dontcheva M., Hertzmann A., Wilson A. G.: MatrixWave: Visual comparison of event sequence data. In *Proceedings of the SIGCHI conference on human factors in computing systems* (2015), ACM, p. forthcoming. 1