# Filter and Compare: Exploring Differences in Event Log Subsets (Poster)

Liliia Aliakberova, Francesca Zerbato, Dirk Fahland
Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, The Netherlands
{l.aliakberova, f.zerbato, d.fahland}@tue.nl

***Abstract*—In exploratory settings, process analysts often begin the analysis with limited prior knowledge of the event log. To build an understanding of the log, they iteratively compute basic statistics, visualize the data and apply filters to narrow their analysis scope. Through filters, analysts can focus on subsets of interest within the event log, which may include cases with delays, specific activities, or other relevant attributes. Filters split the event log into two subsets: a result set, which includes cases that satisfy the filter criteria, and a complement set, including cases that do not. Although the filter reflects the analyst's focus, the analyst is often unaware of what other properties characterize the result set—and, equally important, of what characteristics have been left out. This makes it difficult for the analyst to maintain an overview of the analyzed process. We therefore propose to compare and visualize differences between the result set and its complement to reveal insights that were not part of the original focus but emerged as a result of the filtering. This ongoing research presents this problem in detail and explores dimensions along which differences between a result set and its complement could be detected and presented. Our long-term goal is to support analysts in navigating their exploration.**

***Index Terms*—Process Mining, Exploration, Event Log Subset Comparison, Case Filtering**

## I. Introduction

Process mining (PM) studies processes based on event logs [1]. Event logs contain complex, large-scale process data about how processes work in real-world systems. In *exploratory* settings, analysts often begin their work with limited prior knowledge about the process and must make sense of the data through interactive analysis [2]. Unlike directed analysis, which follows a fixed plan, exploratory PM is emergent, iterative, and driven by the insights that unfold during the analysis. Each step builds on findings from previous steps and analysts refine their questions, assumptions, and strategies as new patterns emerge [3]. This dynamic nature is reflected in established frameworks: the PM$^2$ methodology, which emphasizes the iterative discovery of insights [4], and the Visual Analytics cycle, which describes how data understanding evolves through interaction with visualizations [5].

To build an understanding of the log, analysts iteratively apply filters to narrow their analysis scope, compute basic statistics, and visualize process data in various forms [2], [3]. Through filters, analysts can isolate event log subsets of their interest, for example, delayed cases or cases with a particular activity. Filters split the event log into two subsets: a result set, which includes cases that satisfy the filter criteria, and a complement set, which includes cases that do not.

Although filtering helps narrow the analysis scope, it often leads analysts to focus only on one subset of the log (the result set), while ignoring the subset(s) of cases that have been excluded by the filter. This is also because many PM tools do not show a side-by-side comparison of filtered and excluded subsets, placing the responsibility for this comparison on the analyst [6]. As a result, analysts often lack a complete view of the entire log, which may prevent them from recognizing the characteristics of the result set and, equally important, the nature of what was filtered out. This limited visibility makes it difficult for analysts to interpret filtered data within the broader context of the process, assess how much process behavior was excluded, or validate and refine their filtering strategies [7].

To address this problem, we propose to *compare and visualize* differences between the result set and its complement to reveal insights that emerged as a result of filtering. We believe that visibility into both subsets and facilitation of structured comparisons could help analysts understand what makes a subset relevant, unique, or representative, thereby guiding their exploration. However, finding ways to compare subsets that are "meaningful" to process analysts is per se a significant challenge. This ongoing research presents an approach for comparing a filtered subset with its complement to enhance the exploration of event log data. We introduce the problem through a motivating example in Sect. II. Next, in Sect. III, we outline possible dimensions for comparison, and, in Sect. IV, we describe a prototype to support subset comparison. We conclude by highlighting open directions for future work.

## II. Problem Description

In this section, we introduce the problem by first outlining basic concepts on event logs and case filters, followed by a motivating example that leads to our research question.

### A. Basic Concepts: Event Logs and Case Filters

An *event log* is a structured collection of records that capture the execution of business processes. An *event* $e$ is represented as a set of attribute–value pairs, modeled by a partial function $\pi : \mathcal{E} \times AN \nrightarrow Val$, where $\mathcal{E}$ is the universe of events, $AN$ is a set of attribute names, and $Val$ the domain of possible values. For an attribute $a \in AN$, $\pi(e, a) = v$ assigns a value $v$, or $\pi(e, a) = \perp$ if undefined. Each event must include a timestamp ($\pi(e, time) \neq \perp$), an activity attribute $\pi(e, \text{act}) \neq \perp$, where

act $\in AN$ identifies the activity being executed, and optionally other additional attributes ($\pi(e, a) \neq \perp$, with $a \neq$ time $\wedge\, a \neq$ act) [1], [8]. Events can be grouped into *cases*. A case is a sequence of events denoted by $\sigma = \langle e_1, e_2, \ldots, e_n \rangle$ and uniquely identified by a designated *case identifier* $c \in AN$. An event log $L$ is a set of cases: $L = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$. This structure allows analysts to explore and compare different process executions in a consistently [8], [9]. Filters allow analysts to select subsets of an event log, i.e., sets of cases, that satisfy a specific condition. A case filter $F$ can be defined by one or more *predicates*, where each predicate imposes a logical condition on event attribute values using operators such as $=, \neq, <, \leq, >, \geq$. These predicates can be existential (e.g., $\exists e : e.\text{amount} \geq 2$) or universal. Applying a case filter $F$ to an event log $L$ results in a partition of the log into two disjoint subsets: a **result set** $L' \subseteq L$ which includes all cases that satisfy $F$, and a **complement set** $\bar{L}' = L \setminus L'$, which includes all other cases. By definition, $L = L' \cup \bar{L}'$ and $L' \cap \bar{L}' = \emptyset$.

### B. Motivating Example: Exploring through Filters

As an example, consider a process analyst—let us call her Alice—who explores the *Road Traffic Fine Management (RTFM)* event log [10]. Alice aims to discover scenarios in which offenders do not pay their fines. Guided by this goal, she first applies a case filter $F_1$ to keep cases that do not have a "Payment" activity ($F_1 : \nexists e \,|\, e.a_{name} =$ *'Payment'*). $F_1$ splits the RTFM log, into a result set $L'$ that includes cases without payment and a complement set $\bar{L}'$ that includes cases with payment. However, in her PM tool, Alice sees only $L'$, which she explores further. Initially, Alice examines the presence and frequency of activities within this subset. Although she knows that the cases in $L'$ do not involve payment, she lacks information about the remaining activities, e.g., whether some of them are unique or interesting to explore further. To understand this better, she needs to explore the process control flow, take note of any relevant patterns she discovers, and remove the filter to compare her notes against the entire log. In doing so, she finds an activity called "Insert Fine Notification", which she considers interesting for her analysis. She hypothesizes that if offenders do not receive a notification, they might not be aware that they have a fine to pay. To test this hypothesis that emerged from filter $F_1$, she applies a new filter $F_2 : \exists e \,|\, e.a_{name} =$ *'Insert Fine Notification'* over the whole event log to retain cases with this activity. Alice then inspects the results, again focusing on the control flow and relevant data attributes.

Every time Alice applies a filter, she can only see a subset of cases that meet the filter criteria. This filtered view reveals only part of the process behavior, meaning she has no visibility into what has been excluded. Although she knows the selected cases match the filter, she lacks support to understand how they differ from the rest. Are these cases significantly different in their control flow? Do they show unique activities? Do they vary in data attributes, duration, or outcomes? Moreover, since she is still exploring, Alice is uncertain that the filters she applies (e.g., excluding "Payment" or including "Insert Fine Notification") actually isolate the cases of interest. How can she validate her filter logic without being able to compare the filtered subset with the excluded cases?

Alice's example analysis leads us to the central problem addressed in this work: **How can we compare the filtered subset of an event log with its complement to support analysts in navigating their exploration?** This question narrows our focus to the comparison of two subsets derived from applying a single case filter to an event log. By design, these subsets are disjoint and have the same set of attribute names, which eases their direct comparison.

## III. Dimensions for Subset Comparison

To derive a set of dimensions which suggest differences and similarities between a result set and its complement, we have reflected on typical aspects analysts are interested in when exploring and comparing event logs. More precisely, we have drawn from existing process mining literature [3], [11]–[13], empirical insights into process mining practices [2], and worked with filtering in different PM tools.

In particular, insights into process mining practices helped us identify initial requirements for comparison dimensions and related indicators. Since we aim to help analysts compare subsets as part of their *interactive* exploration process, such indicators must be easily interpretable and allow analysts to understand the differences without requiring specialized expertise. In addition, these indicators should be efficient to compute, allowing interactive exploratory analysis without causing delays. Indicators should also be relevant to typical analysis tasks and the filter applied, e.g., they should capture temporal behavior, performance, or activity patterns. Finally, it is important that the comparison is presented in an intuitive and accessible manner and supports analysts in the identification of patterns and characteristics associated with subsets.

Based on these requirements, we identified a selection of common and well-known indicators, which we organized into the dimensions summarized in Table I. It is important to note that the indicators we propose can be aggregated (for example, using statistical methods or distributions) to facilitate set-level comparisons or used to highlight specific set elements, such as cases with the longest waiting times or rare activity sequences.

TABLE I
Summary of Comparison Dimensions and Key indicators

| **Dimension** | **Key indicators** |
|---|---|
| Subset Composition | Subset size/ratio<br>Variant counts/ratio<br>Temporal distribution |
| Control Flow | Activity presence/frequency<br>Variant differences<br>Looping/skipping behavior |
| Performance | Case duration<br>Activity duration<br>Waiting time |
| Data Attributes | Distribution of values<br>Subset-specific presence<br>Event-/case-level attributes |
| Inter-Attribute Interactions | Attribute-conditioned activity behavior<br>Performance variation<br>Context-dependent control flow |

### A. Subset Composition

The subset composition dimension captures the size, structure, and timeline of the result set and its complement. It can be used to show how each of the two sets compares to the entire event log. This dimension includes indicators such as *subset size and ratio*, which describe the number of cases in each subset and their relative proportions. It also includes *variant counts and ratios*, reflecting the number of distinct case variants present in each subset and their share of the overall process. Furthermore, the *temporal distribution* compares the start and end times of cases between subsets. This dimension can help show the effects of the filter quantitatively.

### B. Control Flow

Control flow describes characteristics associated with activities and their order within cases [1]. This dimension incorporates indicators such as the *presence and frequency of activities*, that highlights if specific activities appear only in one subset, or their frequency differ significantly between subsets [1]. It also includes the *variant differences* indicator, capturing trace variants that occur exclusively in a subset. Furthermore, an indicator related to *looping and skipping behavior* reflects the cases where activities are repeated or bypassed within a subset [1]. Figure 1 illustrates this dimension using

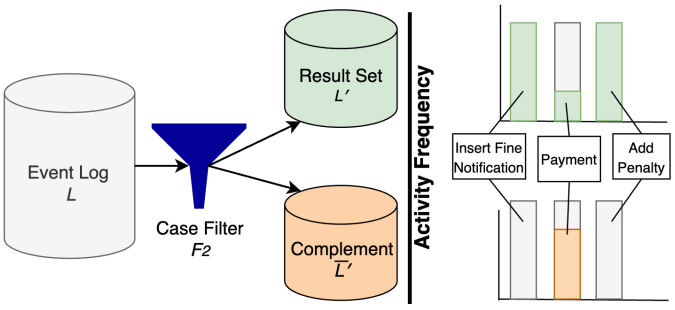


Fig. 1. **(Left)** The split of the log $L$ into a result set $L'$ and its complement $\bar{L}'$generated by a filter, e.g., $F_2$. **(Right)** Comparison of subsets based on *Activity Frequency*. Each bar in the chart represents the number of cases with a specific activity in the full log (grey background); in the result set (green filling), and in the complement (orange filling).

filter $F_2$ applied by Alice to the RTFM log to keep cases with activity *Insert Fine Notification* (see Section II). The bar chart shows that *Add Penalty* appears only in the result set, while *Payment* is more frequent in the complement. These differences, based on filter F2, highlight how activity *Add Penalty*, which was not used in the filter, behaves similarly to *Insert Fine Notification*—something that Alice was not aware of. While simple, this dimension helps analysts examine how the presence, frequency and order of the activity differ between subsets.

### C. Performance

Performance can be assessed from various perspectives, but in this context we focus on it from a time perspective. Time-related performance differences can motivate initial filtering in the context of performance analysis, such as isolating long-running or delayed cases. Comparison of subsets across this dimension includes time-related indicators, such as *case duration*, which measures the time between the first and last event in a case; *activity duration*, defined as the time between the start and end events of an activity when such pairs are available; and *waiting time*, the interval between the end of one activity and the start of the next [1]. The indicators in this dimension can be aggregated on the level of sets, cases or events. This dimension helps detect general trends, bottlenecks, inefficiencies, or delays that distinguish the result set from its complement, highlighting possibly problematic areas for further exploration [1], [14].

### D. Data Attributes

The data attribute dimension captures data attribute information about events or entire cases. These attributes, which are not related to timestamps or activities, can differ between subsets of the event log after filtering. For this dimension, we consider indicators such as the *distribution of attribute values*, which can differ between subsets, and the *subset-specific presence*, where some attributes appear exclusively in one subset, highlighting distinctive characteristics. Event logs contain case-level (static) and event-level (dynamic) attributes. While case attributes are easy to compare, event attributes require aggregation (e.g., most frequent value, sum, first/last value) to be compared. This dimension can reveal patterns that may indicate correlations among data attributes and other process perspectives, such as control flow or time.

### E. Inter-Attribute Interactions (Cross-Dimensional)

This dimension focuses on how different attributes in the event log, such as time, activities, and data attributes, relate to each other. It highlights combinations of attributes that are notably different or similar across the two subsets after filtering. Indicators in this dimension include *attribute-conditioned activity behavior*, which examines how data attributes influence the occurrence or frequency of specific activities; *attribute-based performance variation*, which identifies delays or durations related to particular combinations of attributes; and *context-dependent control flow*, which captures changes in the structure or sequence of activities changes depending on the values of specific data attributes. These comparisons support multi-dimension analysis and may help uncover differences in process behavior.

Together, these dimensions and indicators form a framework for comparing event log subsets derived from filtering. The framework, which is still under development, will allow analysts to identify differences, assess filtering effects, and maintain an overall view of the event log. Dimensions and indicators may need to be adjusted based on the analyst's goal and the data under investigation, and may require different approaches to be visualized effectively.

## IV. Baseline Subset Comparison Tool

To explore the dimensions and indicators discussed in Sect. III, we have implemented a baseline tool in Python to compare subsets of an event log along selected indicators.

The tool includes several core components organized into a modular architecture. The *Event Log Loader* allows to upload event logs in standard formats such as CSV or XES. The *Attribute Mapper* enables users to define the core columns of the event log, including caseID, activity, and timestamp. The *Filtering Module* supports the creation and application of filters to isolate subsets of interest. Filters can be saved to help the analyst maintain an exploration pipeline in the *Exploration Tracker* module. The *Subset Comparison Engine* automatically compares subsets (result and complement). The *Visualization Module* provides basic visualizations for each dimension and allows users to choose what subsets to compare (the original log, the result set or the complement). Finally, the *Exploration Tracker* maintains a history of applied filters and generates summary reports containing statistics from the exploration process.

To illustrate the tool, we present its architecture in Figure 2, which also shows the application of the filter $F_2$ by Alice to the RTFM event log [10] to separate those cases that contain the activity *Insert Fine Notification* (see Section II). The aim

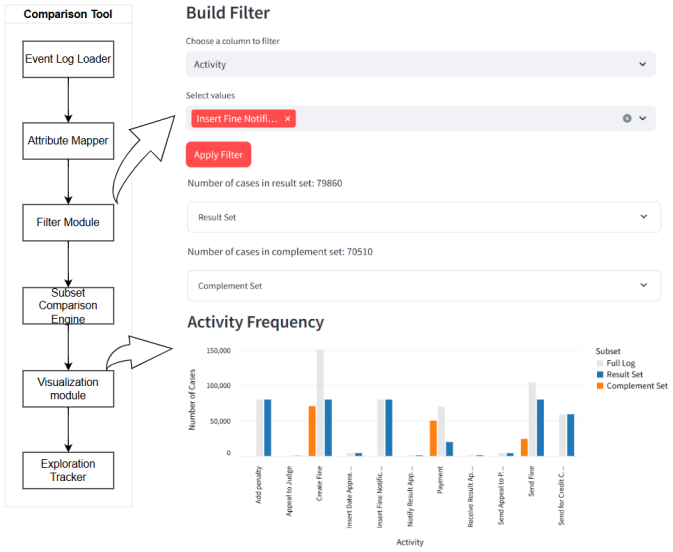


Fig. 2. **(Left)** The architecture of the tool. **(Right)** Filter module showing the application of filter $F_2$ on the RTFM event log, along with the visualization module used to compare the subsets based on *Activity Frequency*.

of this prototype is to explore how we can best provide baseline support for each comparison dimension. It currently implements one indicator per dimension, except for inter-attribute interactions dimension, which is not yet supported.

## V. Conclusion and Future Work

In this poster paper, we have introduced a challenge in exploratory PM that stems from analysts focusing only on the result set and overlook the characteristics of its complement after filtering. To address this, we proposed comparing event log subsets across several dimensions to better understand both the result set and its complement after filtering. In addition, we developed a simple baseline tool that allows analysts to interactively compare event log subsets across these dimensions. While this prototype is basic, it provides a starting point for multi-perspective subset comparison, which is our main aim for future work. Unlike traditional comparative PM approaches [12], which typically focus on comparing different processes or variations of the same process across organizations, our approach emphasizes the role of the complement set. It is specifically designed to support iterative, filter-driven exploration, by providing analysts with cues that help them learn about both the included and excluded sets of cases. In the future, we plan to improve our framework by developing difference measures that automatically detect and highlight differences between subsets that might be interesting to analysts. The future improvements for the tool may include more advanced visualizations and full support for multi-dimensional attribute interactions. We also intend to provide visual cues and allow analysts to interact with their own analysis history to refine their analysis steps. Indeed, as analysts typically apply several filters in sequence, resulting in multiple subsets at different stages, we aim to support comparison between any of these subsets to reflect the evolving nature of exploration. We also plan to evaluate our approach on real event logs and conduct user studies with PM analysts to evaluate its usefulness in helping analysts better explore event logs.

## References


[1] W. M. P. van der Aalst, *Process Mining: Data Science in Action*, 2nd ed. Cham, Switzerland: Springer, 2016.

[2] F. Zerbato, P. Soffer, and B. Weber, "Initial insights into exploratory process mining practices," in *BPM Forum*, ser. LNBIP, vol. 427. Springer, 2021, pp. 110–125.

[3] C. Klinkmüller, R. Müller, and I. Weber, "Mining process mining practices: An exploratory characterization of information needs in process analytics," in *BPM*. Springer, 2019, pp. 322–337.

[4] M. L. van Eck, X. Lu, S. J. J. Leemans, and W. M. P. van der Aalst, "PM$^2$: A process mining project methodology," in *Advanced Information Systems Engineering (CAiSE 2015)*, ser. LNCS, vol. 9097. Springer, 2015, pp. 297–313.

[5] D. A. Keim, F. Mansmann, J. Schneidewind, J. Thomas, and H. Ziegler, "Visual analytics: Scope and challenges," in *Visual Data Mining: Theory, Techniques and Tools for Visual Analytics*. Springer, 2008, pp. 76–90.

[6] M. Salas-Urbano, C. Capitán-Agudo, C. Cabanillas, and M. Resinas, "LoVizQL: a query language for visualizing and analyzing business processes from event logs," in *Int. Conf. on Service-Oriented Computing*. Springer, 2023, pp. 13–28.

[7] F. Zerbato, M. Franceschetti, and B. Weber, "A framework to support the validation of process mining inquiries," in *BPM Workshops*, ser. LNBIP. Cham: Springer, 2024, vol. 249, pp. 249–266.

[8] D. Fahland, "Extracting and pre-processing event logs," *arXiv preprint*, vol. arXiv:2211.04338, 2022.

[9] W. M. P. van der Aalst, "Process mining in the large: A tutorial," in *LNBIP*, 2014, vol. 172, pp. 33–76.

[10] M. de Leoni and F. Mannhardt, "Road traffic fine management process," *Eindhoven University of Technology, Dataset*, vol. 284, 2015.

[11] A. Del-Río-Ortega, M. Resinas, C. Cabanillas, and A. Ruiz-Cortés, "On the definition and design-time analysis of process performance indicators," *Information Systems*, vol. 38, no. 4, p. 470 – 490, 2013.

[12] W. M. P. van der Aalst, S. Guo, and P. Gorissen, "Comparative process mining in education: An approach based on process cubes," in *Data-Driven Process Discovery and Analysis*. Springer Berlin Heidelberg, 2015, pp. 110–134.

[13] A. Bolt Iriondo, "Comparative process mining: analyzing variability in process data," Phd Thesis 1 (Research TU/e / Graduation TU/e), Mathematics and Computer Science, Jan. 2023, proefschrift.

[14] S. Suriadi, C. Ouyang, W. M. P. van der Aalst, and A. H. M. ter Hofstede, "Event interval analysis: Why do processes take time?" *Decision Support Systems*, vol. 79, pp. 77–98, 2015.