Visualization Methods to Support Real-time Process Monitoring

Zsuzsanna Nagy [1] and Ágnes Werner-Stark[1]

[1] *University of Pannonia, Egyetem st. 10, Veszprém, 8200, Hungary*

Abstract

The industrial sector has a keen interest in solutions that monitor ongoing process executions and swiftly detect deviations from desired behaviours, thereby facilitating timely corrective interventions. Visualizing complex process data offers an intuitive and comprehensive perspective, enabling stakeholders to quickly identify deviations and their root causes. Through conformance checking techniques, deviations are automatically detected and their severity is estimated. In this work, two visualization methods for process data are proposed: one for original event data and another for the outputs from an alignment-based multi-perspective online conformance checking technique, termed alignment data. The innovative aspect of these methods is their ability to support both fixed and duration timeline-based visualizations for event and alignment data. They also adeptly handle overlapping objects while making it possible for the users to select the perspective from which they want to examine the data. When tested on a real-life manufacturing process, our methods identified timing anomalies and consistent product quality-affecting deviations, emphasizing the crucial role they play in better process oversight.

Keywords

Process mining, visualization, visual analytics, conformance checking, multi-perspective

1. Introduction

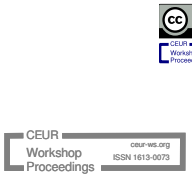
Process mining is a family of techniques that help to extract knowledge from process data in order to discover, monitor and improve business processes [1]. Conformance checking, a subset of process mining, evaluates how well the observed process executions align with a predefined process model, by identifying deviations and measuring their severity. Currently, the de facto standard technique for conformance checking is the calculation of alignments [2], which provide a path for each process instance through the process model. The goal is to find the optimal alignments, which minimize the total cost of the deviations. An alignment is basically a sequence of (alignment) moves, where a move consists of a pair of an event from the process instance and a transition from the process model.

The basic alignment-based conformance checking technique is limited to working offline, focusing only on completed cases, and considering just the control-flow perspective (i.e., the ordering of events). For effective real-time process monitoring, the technique should also accommodate ongoing cases, enabling the detection of deviations before a process instance concludes and allowing for timely corrective interventions. Furthermore, since deviations can arise from various perspectives (e.g., time, resource, or other data), it is essential for the technique to account for these as well. For these reasons, in this paper, we focus on the alignment-based multi-perspective online conformance checking (MOCC) technique and its outputs, which were introduced in one of our previous works [3]. The outputs are called multi-perspective prefix-alignments, but for simplicity, we will refer to them as alignment data in most of the remaining part of this work.

Visualization is crucial for process monitoring, because it provides an intuitive and comprehensive representation of complex process data, enabling stakeholders to quickly identify deviations and their

Proceedings ITTAP'2023: 3rd International Workshop on Information Technologies: Theoretical and Applied Problems, November 22–24, 2023, Ternopil, Ukraine, Opole, Poland

EMAIL: nagy.zsuzsanna@mik.uni-pannon.hu (A. 1); werner.agnes@mik.uni-pannon.hu (A. 2)
ORCID: 0009-0009-5413-7284 (A. 1)



root causes. Incorporating temporal information into visualizations is essential, as it offers context, sequence, and duration to events, fostering a deeper understanding of dynamics, dependencies, and progression over time. There are various process mining software tools available, yet the existing visualization techniques for event data do not properly represent temporal information. Events with the same timestamp, known as overlapping events, are particularly problematic and often not handled effectively. Additionally, events are typically represented either as time points or time intervals, with no methods capturing both simultaneously. While a few visualization methods exist for the outputs of conformance checking, none incorporate temporal information. Consequently, visualizing deviations highlighted by conformance checking algorithms remains an open challenge [4]. A detailed overview of existing solutions is provided in the next section.

The visualization methods for process analytics can be divided into three main categories according to the input data and the information to be visualized: process model visualization, event data visualization and linked visualization [5]. This paper aims to develop a visualization method for the original observed event data obtained from an event stream, as well as a method for visualizing outputs of the MOCC technique, the alignment data. This alignment data is calculated from event data and the associated process model. The former falls under the event data visualization category, while the latter fits into the linked visualization category.

Both of the proposed visualization methods were designed to encompass the beneficial aspects of existing methods while addressing their limitations. The innovative aspect of them is their ability to support both fixed and duration timeline-based visualizations for event and alignment data. Additionally, they adeptly handle overlapping objects while making it possible for the users to select the perspective from which they want to examine the data.

The proposed visualization methods were implemented within a test environment. Their utility and performance were evaluated using simulated event streams from a real-life process. The experiment revealed that our methods effectively identified timing anomalies, primarily linked to machine downtimes, as well as other issues potentially stemming from data quality problems or malfunctioning equipment. These findings underscore the urgent need for timely interventions.

The remainder of this paper is structured as follows: Section 2 discusses related works, while Section 3 introduces the basic concepts necessary for understanding the proposed visualization methods. Section 4 provides a detailed description of these methods. Finally, Section 5 evaluates these methods and presents the results of the experiments. Finally, the paper is concluded.

2. Related works

In this section, existing process mining software tools are reviewed, followed by an analysis of the visualization methods used by them for event data and the outputs of conformance checking algorithms, including alignments.

2.1. Process mining tools

There are various process mining software tools available. In [6], a comparison was made of the performance and features of four most popular tools: ProM, Disco, Celonis, and My-Invenio. Reference [7] lists 27 available tools, but detailed examination was done only on the previously mentioned four tools, with the addition of Apromore. The most extensive analysis can be found in [8], where 17 tools were evaluated in depth. Additionally, a website has been developed that hosts a continually updated list of existing process mining software tools, offering a platform for comparative analysis.

According to the findings in [8], the examined tools are limited in their capability to handle timestamps; they can manage event data with either one or two timestamps, but not both. In terms of process discovery, certain tools can display process execution variants (also termed trace variants) and provide a detailed view of a specific case. While these are valuable for visualizing the range of observed process executions, they do not offer contextual information. For example, they do not illustrate the number of ongoing cases, the activities being performed at any given time, the resources involved, or their locations. When it comes to conformance checking, some tools can present an aggregated view of deviations overlayed on the process model. Additionally, they can list deviations (both undesired and

missing activities) as well as non-compliant activity sequences. However, this representation is often limited to textual or tabular formats.

2.1.1. Tools built on PM4Py

A primary limitation of the tools examined in [8] is that, with the exceptions of Apromore and ProM, they are not open-source. This restricts the possibility of introducing new features. Consequently, many advanced and recent process mining algorithms are not incorporated into these tools. In contrast, PM4Py [9] offers an open-source process mining library for Python, which boasts easy extensibility. Numerous process mining applications have been developed using this library as a foundation.

In [10], an approach was introduced that identifies bottlenecks in business processes by analysing event logs and presenting the results visually. This method can represent activity durations and trace durations over time as dot charts. However, a limitation of this solution is its emphasis on summarized values at the expense of insights into individual traces, even though it values temporal information.

In [11], a process mining toolkit named PM_{TK} is introduced that is able to show the trace variants (i.e., the distinct process execution patterns) and visualize events over time using a dotted chart, similarly to ProM [12]. However, it cannot perform conformance checking.

In [13], a modular software system named Smyrida is introduced, which can also show the trace variants and their frequency, but cannot show them on a timeline. It can perform conformance checking, but the alignments are shown only in tables.

In [14], an interactive process discovery tool named Cortado is introduced. While it can perform conformance checking and visualize alignments, it has a limitation in displaying alignment moves. Specifically, when an alignment is projected onto a trace (or variant), the model moves are not displayed, and vice versa. When projected onto the process model, the log moves are not displayed. Additionally, while it projects statistical measures like service time and waiting time lengths (e.g., minimum, maximum, mean, etc.) onto variants and the model, there is no provision to compare these with expected values.

2.2. Visualization methods for event data

When visualizing event sequences for online representation, we focus exclusively on timeline-based visualizations. Such visualizations graphically portray sequences of events, aligning them on a temporal axis to convey the order of events within event sequences [15].

For event sequence data, timeline-based visualizations can be categorized based on their way of representation of time: fixed timeline-based (like dotted charts [12]), duration timeline-based (including Gantt charts, bar charts, and their variants [16]), and a hybrid of both (as demonstrated in our previous work [17]). Typically, individual events are represented based on their timestamp(s) and another chosen attribute, often the case-id or resource. These events manifest as graphical objects (like circles or rectangles) and are color-coded based on the activity.

Existing visualization techniques, however, struggle with a significant challenge: the handling of overlapping events. When multiple events share identical temporal data, one will cover the others in the visualization. Furthermore, in fixed timeline-based visualizations, when an activity execution spans multiple events (for example, a start event followed by a completion event), these events are not linked. This omission fails to convey the activity's execution duration.

2.3. Visualization methods for alignment data

When it comes to visualizing the outputs of conformance checking techniques, including alignment data, no timeline-based methods currently exist. Consequently, our discussion encompasses all existing methods.

Reference [18] offers a thorough examination of how conformance checking results are visualized by various process mining tools. Addressing the “Why?” aspect, the study delved into the structuring of visualizations within the tool interfaces, categorizing them into four primary groups:

1. Quantify conformance,
2. Break-down and compare conformance,
3. Localize and show deviations,
4. Explain and diagnose deviations.

Our primary focus is on the "Localize and show deviations" category, which features methods that present alignments for specific process variants. These methods predominantly use color-coded flowcharts and chevron diagrams (sequences of coloured arrowheads) to depict alignments of all activities within a trace.

Visualization methods based on chevron diagrams are available for both control-flow alignments [19] and multi-perspective alignments [20]. Both techniques employ coloured chevron diagrams to portray alignment moves in sequence, with arrow colours signifying move types. In multi-perspective alignments, there is a variation where arrowheads are coloured according to activity, and move types are depicted as coloured bars above the arrows.

Although these methods provide significant insights, they are primarily suited for offline contexts due to the absence of temporal information representation. Nonetheless, the color-coding schemes for different alignment move types remain relevant for online contexts.

3. Background and preliminaries

In this section, fundamental concepts tied to the input data (event and alignment data) are explained to enhance understanding of the proposed visualization techniques.

3.1. Event data

Every process mining algorithm takes event data as its input. This event data, a subset of broader process data, captures the observed behaviour of executed processes. In offline settings, this data typically comes in the form of an event log, while in online settings (which is our case), it is presented as an event stream. Each event must possess a case identifier and a recorded activity. The case identifier specifies the context of the executed activity. Additionally, events can have other attributes providing more details about the activity, such as the person or machine performing the activity (i.e., resource) or the time of execution (i.e., timestamp).

The execution of an activity, often termed an "activity instance", may consist of multiple events. Each of these events describes different lifecycle transitions of the activity. A comprehensive list of possible transitions can be found in [21]. The most common transition types are "start" and "complete", denoting the beginning and the end of an activity instance's execution, respectively. If the same activity can occur multiple times within the same context, the event should also include an activity instance identifier. The sequence of events executed within the same context is termed a "trace".

Table 1
Content requirements for an observable unit in different event streams

Attribute	Single event stream	Composite event stream
case identifier	required	required
activity	required	required
activity instance identifier	required	irrelevant
transaction type	required	irrelevant
timestamp	one	two (start and complete)
other attributes (e.g., resource)	optional	optional

In this paper, it is assumed that an event stream is composed of observable units from which process information can be extracted. Event streams are categorized into two types based on the number of events each observable unit corresponds to: the single event stream and the composite event stream. In a single event stream, each observable unit contains information about a single event of an activity

instance. Conversely, in a composite event stream, each observable unit encompasses details of all the events within an activity instance, yet only includes the timestamps for the start and complete events. Table 1 summarizes the attributes for which values are present within observable units for each type of event stream. Observable units may also encompass additional attributes. The sequence of attribute values within an observable unit is flexible, but it must remain consistent throughout.

3.2. Alignment data

The outputs of alignment-based conformance checking algorithms are (prefix-)alignments, that are derived from event data and a predefined process model. They serve to contextualize traces within the process model by mapping them to an execution sequence of that model. Specifically, an alignment consists of two rows:

1. The first row represents a trace (a sequence of events).
2. The second row corresponds to a firing sequence from the process model.

Each row essentially depicts a sequence of moves, either in the log (for traces) or the process model. The key distinction between alignments and prefix-alignments lies in the assumed state of the process execution. For alignments, the process execution is deemed complete, indicating that the trace has reached its end. In contrast, prefix-alignments operate under the assumption that the process execution is ongoing. This means the trace is incomplete, representing just a prefix of the potential full sequence.

In alignments that consider only the control-flow perspective (i.e., the sequence in which activities occur), there are three types of moves:

- Synchronous Move: This indicates an activity that aligns correctly with the model.
- Log Move: This points to an activity that is present in the log but not aligns with the model, signifying an inserted activity.
- Model Move: This can represent either a skipped activity (if the activity is observable) or an unobservable activity.

When other perspectives come into play, synchronous moves are further categorized into:

- Correct Synchronous Move: This signifies an activity that not only aligns with the model, but also has the correct attribute values, a perfect alignment.
- Incorrect Synchronous Move: This indicates an activity that aligns with the model, but has attribute values that deviate from the expected values.

In this paper, it is assumed that the alignment data consists of multi-perspective alignment moves. Whether an alignment move is part of a prefix-alignment or complete alignment, in terms of visualization, it does not matter. An alignment move contains the original event data, if it is not a model move. For model moves, the necessary missing attribute values are obtained from adjacent alignment moves. For incorrect synchronous moves, the name of the corrected attributes and the corrected values are stored separately from the original values.

4. Proposed techniques

We developed our visualization techniques based on Tamara Munzner's visual analytics framework, which examines three key aspects: "What?", "Why?", and "How?" in relation to visualization [22].

The "What?" aspect pertains to the data available for visualization. Our data sources include the event data extracted from an event data stream, as well as the outputs of the MOCC technique. The latter (i.e., alignment data) is derived from both the event data and the predefined process model.

The "Why?" aspect defines the main purpose of a visualization, as well as any potential secondary purposes. In this context, the primary objective of both visualizations is to assist in monitoring process executions. The event data visualization offers a comprehensive view of the real process, while the alignment data visualization highlights and exhibits any deviations from the expected process. When used jointly, they facilitate the rapid detection of deviations and the identification of their root causes, enabling users to take corrective measures.

The “How?” aspect defines the design choices to visualize the data. Based on insights from the literature review, we aimed to make design decisions that encompass the beneficial aspects of existing methods while addressing their limitations.

In this section, the specifics of the proposed visualization methods are introduced. Shared characteristics will be discussed first, followed by comprehensive explanations of each method, which are supported by clear examples to enhance understanding.

4.1. Common properties

Several methods have emerged to display overlapping events over time [23]. Given the potential for multiple overlapping events, two existing techniques stand out as suitable: vertically listing events in parallel (akin to a Gantt chart) with an option of either overlaying them partially or not. Although overlays might condense the chart's size, they could hinder clarity. Consequently, we opted against overlapping visual elements. Thus, both the event data and the alignment data are presented using a Gantt chart-inspired mixed (both fixed and duration) timeline-based visualization.

On the y-axis, a custom attribute is used. If the resource is provided, it becomes the default attribute; if not, the case identifier is used. Importantly, users can select any attribute that is present in every event. Depending on the space required to visualize the objects without overlap, a single attribute value might be represented across multiple rows.

Visualized objects, such as events and alignment moves, are denoted by small vertical lines (“|”). Events and alignment moves pertaining to the same activity instance are connected with bars. Notably, a single event stream's observed unit is regarded as a single event, while a composite event stream's observed unit is depicted as two distinct events (start and complete).

For increased interactivity, users can filter displayed events and alignment moves based on one selected attribute and its selected values, which is ideal for analysing the performance of specific resources, for instance. Hovering over objects reveals relevant information via tooltips and connects related objects by their case identifier in observed order. These interactive features grant users deeper insights without overwhelming the main display.

Distinctive features between the two visualization methods lie primarily in colour choices and tooltip content. These distinctions are elaborated upon in the following subsections. As examples, a trace from a manufacturing process consisting of eight sequential operations (labelled “ts1o”, “ts2o”, etc.) is referenced. In this process, errors (labelled “terr”) can manifest at any stage. A more in-depth description of this process can be found in Section 5.2.

4.2. Visualization of event data

In the visualization of event data, the objects representing the events and their connections are coloured based on the value of the activity attribute. The colours are assigned to the values from a predefined categorical colour scale in the order new values are observed.

When the mouse is hovered over the object representing an event, the tooltip displays all original information from the observed unit, organized as pairs of attribute names and values on distinct lines. The timestamp is prioritized, appearing on the first line, but only showcasing the time value, since events from the same date are displayed. For observed units from a composite event stream, the tooltip of the start event object shows the start timestamp, while the complete event object shows the complete timestamp of the activity instance.

In Figure 1, a visualization of event data for a single case is presented, with the custom attribute set as “nest”. The cursor hovers over the event labelled “ts8o”, representing operations at station 8. Even before conducting conformance checking, several deviations from the expected behaviour are evident. The sequence of events is incorrect (for instance, “ts1o” is succeeded by “ts4o” instead of “ts2o”), and the duration of the activity “ts4o” is unusually extended. Moreover, since the product was identified as faulty before reaching station 5, it should have been discarded by station 7. However, it proceeded to station 8.

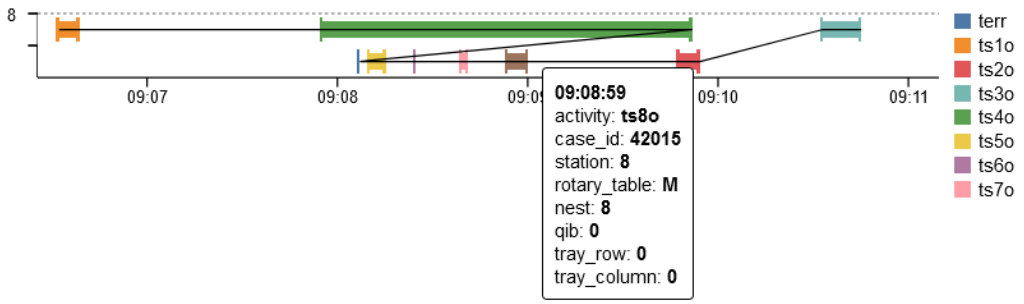


Figure 1: Event data visualization for a single case in a manufacturing process

4.3. Visualization of alignment data

In the visualization of alignment data, specifically multi-perspective prefix-alignment moves, objects representing alignment moves and their connections are coloured based on the value of the alignment move type attribute. This attribute, generated by the MOCC technique, is stored as a numeric value for simplicity. Colours are mapped to these numeric values using a predefined categorical colour scale. The numerical values, alignment move types, and corresponding colours are as follows:

1. Correct synchronous move: green;
2. Incorrect synchronous move: orange;
3. Log move: red;
4. Model move: purple.

When the mouse is hovered over the object representing an alignment move, the tooltip displays all original information from the observed unit, similarly to the event data visualization. If there are corrected values suggested by the process model, they are displayed next to the original values and the whole row is highlighted in red.

All alignment moves are displayed using the original recorded value from the associated observed event, even if deemed incorrect by the process model. The exceptions are process moves, which lack corresponding observed events. In such cases, the recorded values for these moves can be represented by a time value provided by the model. If the model does not specify temporal information for an activity, this time value is set to the (complete) timestamp of the previous alignment move, or if unavailable, the (start) timestamp of the next alignment move. Given that alignments are only computed with at least one observed event for a case, this approach ensures a time value is consistently available for process moves. Similarly, if the model does not provide a value for the attribute displayed on the y-axis, the aforementioned approach is applied.

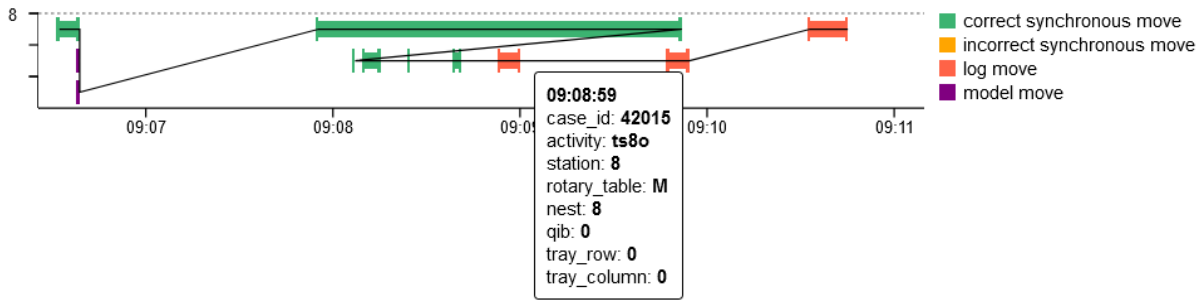


Figure 2: Alignment data visualization for a single case in a manufacturing process

In Figure 2, an example of the visualization of multi-perspective prefix-alignment moves for a trace of a single case is presented, where the custom attribute is “nest”. The cursor is hovered over the alignment move with activity called “ts8o”. This alignment move is a log move, because according to the model, the product should have been discarded at station 7. Notably, the numbers indicating the row and column position of the tray (where the final product is placed) are zero. This could imply that either the product was not positioned on the tray or there was an error in detecting or recording these numbers.

The more significant issue is that the temporal information for “ts2o” and “ts3o” is incorrect, as can also be observed in Figure 1. They were observed after the product reached the last station, instead of after “ts1o”. This is why there are two model moves after “ts1o” and two log moves at the end of the alignment.

5. Results

The proposed visualization methods were implemented within a test environment to evaluate their utility using simulated event streams of a real-life process. In this section, the implementation is first described, followed by an introduction to the process model and event data that are utilized. The results are discussed in the final subsection.

5.1. Implementation

To test the proposed visualization methods, we created a test environment in form of a web application. The backend consists of a server and a middleware and the frontend of a client. The server and the middleware were implemented in Python. The middleware uses a modified version of the newest version of PM4Py library [9] and Flask framework along with Flask-SocketIO library. The PM4Py library was modified based on our previous implementation of the MOCC technique [3], to be able to calculate optimal multi-perspective prefix-alignments. The client was implemented using web technologies (HTML, CSS and JavaScript). The D3 (or D3.js) JavaScript library is used to create and update the visualizations and the Socket.IO JavaScript library is used for the communication between the middleware and the client. The server communicates with the middleware using serialized JSON messages over TCP sockets to transmit data. The middleware sends data in form of JSON messages to the client in real-time using WebSockets facilitated by Flask-SocketIO.

The main task of the server is to simulate an event stream based on the given event log and settings. The settings contain the path to the event log (in csv format) and to the process model (in pnml format), the type of the event stream (single event stream or composite event stream), the number of the column that contains the attribute value for each required attributes of the selected event stream type, the value type for each attribute, and the format of the timestamps.

The middleware observes the event data provided by the server, then processes it and preforms analysis on it. Finally, it send the outputs to the connected clients. When a new client connects, the middleware sends all the necessary information for the visualization and all the cached data that needs to be visualized.

The visualizations are generated and displayed on the client side, based on the data provided by the middleware and the settings given by the user. The two visualizations are displayed on a single page, allowing the user to examine both simultaneously. The user can zoom in, zoom out, reset the zoom, scroll a little left or scroll a little right in both visualizations concurrently.

5.2. The process model and the event data of the examined process

The implemented visualization methods were tested on real-life datasets from manufacturing processes, where one case describes the assembly process of a single product. The process is represented in Figure 3 as a data Petri net (DPN). The original figure, along with the guard expressions for the transitions of the DPN, is available in our previous work [3]. Blue and red transitions correspond to observable activities. In contrast, black transitions do not have any corresponding observable activities. Each variable represents an attribute of the event associated with a given activity. The arrows indicate the variable writing operation. A transition can only fire if all the variable values (both current and new) satisfy the criteria established by its guard expression, presented in the form of a logical formula.

The manufacturing process under discussion is an automated assembly process comprising 8 sequential station operations, labelled “ts1o”, “ts2o”, and so forth. Errors, denoted as “terr”, can arise at any point during the process. When an error surfaces, a three-digit error code is logged, pinpointing both the station and the specific problem type as identified by the machine. For example, the error code

"701" signifies a type 1 error at station 7. Once a product is deemed faulty by the machine, no additional operations are conducted on it. The quality indicator bit, abbreviated as "qib", captures whether station operations were successful (denoted by "1") or unsuccessful (denoted by "0").

Products are assembled concurrently on rotary tables. The main rotary table, denoted as "M", utilizes 8 nests (or product holders), while the smaller rotary table at station 7, labelled "S", has 4 nests. At the final station, products that meet quality standards and faulty products with error codes above 700 are segregated onto separate trays. All other defective products are discarded into a box at station 7. The respective row and column positions where a product is placed on a tray are recorded under the attributes "tray_row" and "tray_column".

In the original event logs, one full trace is stored in every line. There is a start timestamp and duration value (in milliseconds) recorded for every station operation. Since, the values of the attributes only recorded after the execution of the station operations are finished, the log was transformed to be able to be used in form of a composite event stream.

In the original event logs, each line contains a complete trace. For every station operation, a start timestamp and a duration value, measured in milliseconds, are recorded. Because attribute values are logged only after the completion of station operations, the logs were transformed into a composite event stream format. Table 2 displays the attributes found in the resulting files, accompanied by the value type and, where relevant, the value range or format. The order in which they appear in the table is the same as in the files.

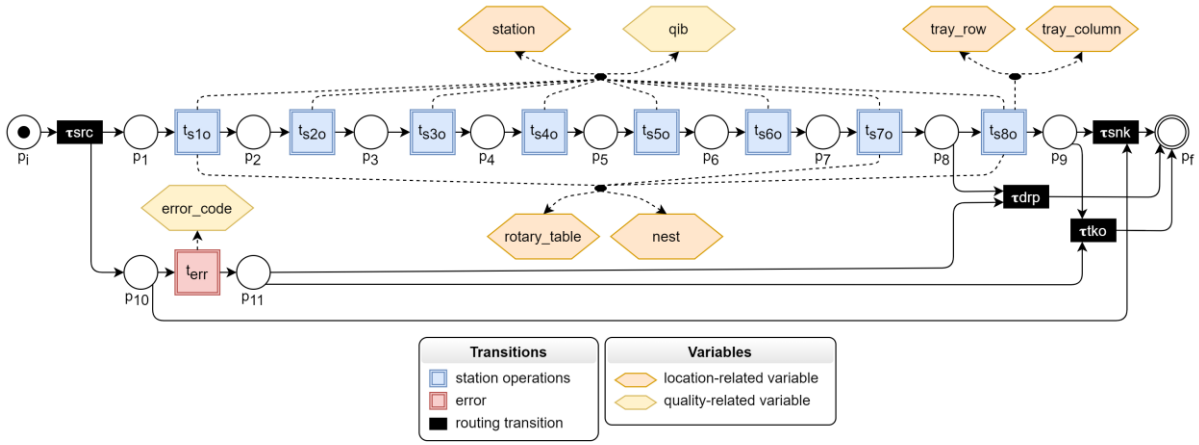


Figure 3: DPN process model of the examined manufacturing process

Table 2
Description of the attributes in the used event logs

Attribute name	Value type	Value range or format
case_id	integer	-
activity	string	["ts1o", "ts2o", …, "ts8o", "terr"]
start_timestamp	datetime	%Y-%m-%d %H:%M:%S.%f
complete_timestamp	datetime	%Y-%m-%d %H:%M:%S.%f
station	integer	[1, 2, …, 8]
rotary_table	string	["M", "S"]
nest	integer	[1, 2, …, 8]
qib	integer	[0, 1]
error_code	integer	[100, 101, …, 899]
tray_row	integer	[0, 1, …, 10]
tray_column	integer	[0, 1, …, 10]

For testing purposes, two types of event logs were generated: one containing only traces with deviations and another capturing the traces from a single day of production. The first set of traces helps

to demonstrate how different deviations are represented in the visualizations. On the other hand, the second set provides valuable insight into how the visualizations appear when applied in a real-life scenario.

5.3. Visualization results and interpretation

In this subsection, the results of the visualization techniques used for the process model (as shown in Figure 3) and the accompanying two event logs are presented and interpreted.

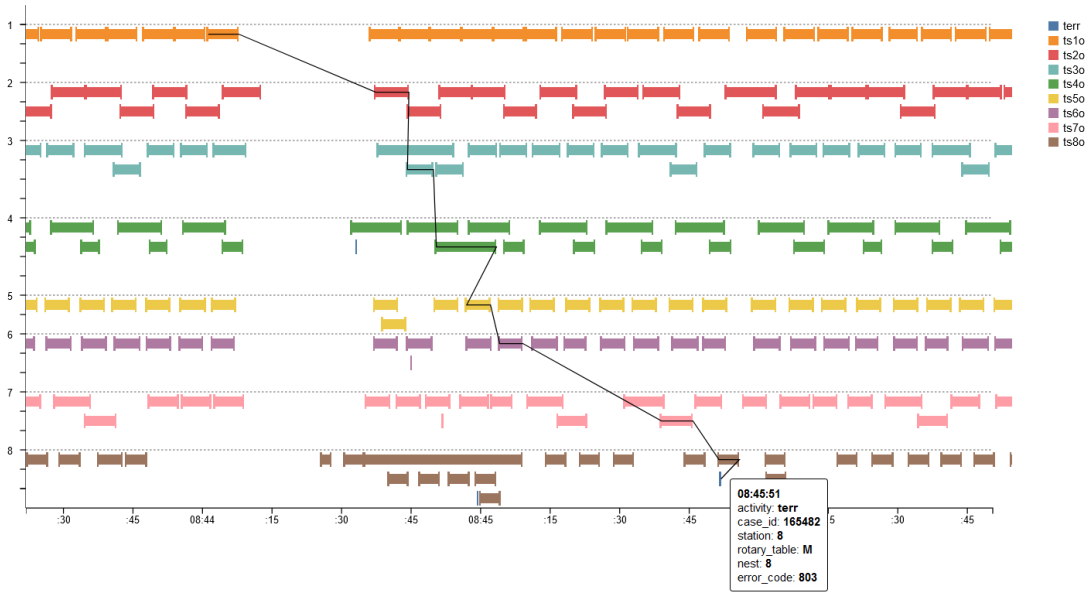


Figure 4: Visualization of event data of single day production traces

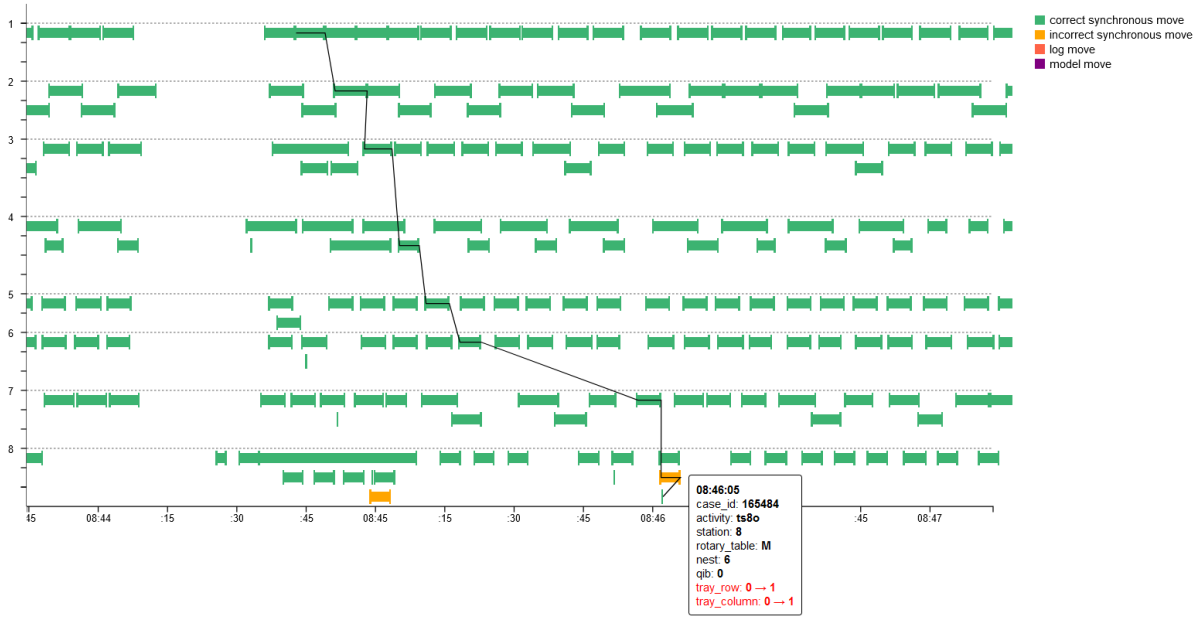


Figure 5: Visualization of alignment data of single day production traces: the most common deviation

A portion of the visualization for event data from single day production traces is illustrated in Figure 4, with "station" as the custom attribute. Ideally, there should not be any overlapping events because the machine can only operate on one product at a time at each station. This suggests that issues might arise from how the durations of the operations are measured or recorded. For some stations, like stations 5 and 6, the time values are mostly accurate, with only occasional overlapping due to brief machine

downtimes. However, at other stations, such as stations 2-4, overlaps are more frequent. Notably, every seventh event at station 3 and every second event at station 4 overlap with their preceding events. Another timing anomaly typically emerges after machine downtimes. For instance, a particularly extended event is observed at station 8, spanning from around 8:44:30 to 8:45:15, which is roughly equivalent to six regularly-timed events. Conversely, at stations 4-7, an unusually short event is noted at each station immediately after the downtime.

Apart from causing timing anomalies, downtimes seem to increase the number of faulty products as well. In Figure 4, a process execution interrupted by a brief downtime is highlighted, leading to a faulty product at the end. However, this is not the sole instance. Two additional errors are observed: one at station 4 around 8:44 and another at station 8 around 8:45, resulting in two more defective products.

A portion of the visualization for alignment data from single day production traces is illustrated in Figure 5 and Figure 6, using “station” as the custom attribute. It is evident that the majority of traces fully align with the model, with only a few deviations from expected behaviour. Although these deviations are infrequent, their consistent appearance can lead to significant challenges over time. Notably, most of these deviations occur at the final station, station 8. In Figure 5, an example of the most common deviation type is highlighted. A final product is either not placed on the tray or the “tray_row” and “tray_column” values are not recorded. The process model selected a value of one for both attributes, as the sole criterion defined within the model is that these values cannot be zero. In Figure 6, an example of the second most common deviation type is highlighted. A semi-finished product, declared defective at station 2, was not discarded at station 7 but instead reached the final station. With both “tray_row” and “tray_column” values at 0, this mirrors the situation in Figure 5, suggesting the product either was not placed on the tray or the values were not recorded.

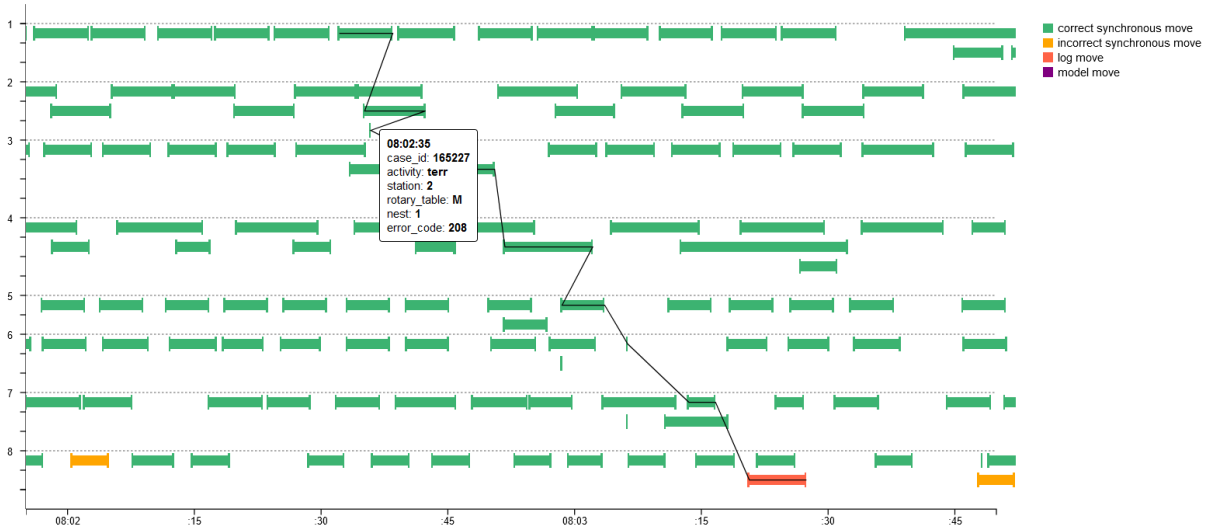


Figure 6: Visualization of alignment data of single day production traces: the second most common deviation

When we applied the methods to solely deviating traces, our observations mirrored those previously discussed:

- One or several downtimes during a process execution resulted in events that were either too short or too long, leading to the product becoming faulty, or at the very least, being declared faulty by the machine.
- Semi-finished products that were declared faulty were carried to station 8 instead of being discarded at station 7.

Regardless of the underlying reasons, it is crucial to address the identified issues to ensure that all products, especially the good ones, are handled appropriately. Although the root causes of these deviations cannot be definitively determined based solely on the visualizations, they do help narrow down possible explanations. For instance, assembly line operators could easily verify in real life whether products are actually placed on the tray. If they are, then the issue might merely be a data quality problem. If not, the machine may be malfunctioning and would require repair.

5.4. Performance evaluation

In this subsection, the performance of the visualization methods when applied to the event log with a single day of production is evaluated.

The responsiveness of the visualization methods primarily depends on the number of events or alignment steps displayed. The number of alignment steps is either equal to or exceeds the number of events, based on the total number of model moves. In the examined process instances, model moves were infrequent. Thus, for this discussion, both numbers are considered to be equivalent.

When using a single event stream as input, the number of events matches the number of observed units. In contrast, with a composite event stream, the number of events is approximately double the number of observed units. This is because an observed unit is represented by two separate events in the visualization when the start and complete timestamps differ. In the studied process instances, typically only error occurrences (i.e., "terr" events) possessed single timestamp values. While there are occasional timing anomalies causing station operations to also have single timestamp values, as discussed in the previous section, these instances are infrequent. Consequently, for the purpose of the discussion, it is assumed that the number of events is roughly double the number of observed units.

The optimal number of events appears to be around 1,000. In the processes that were examined, this corresponds to approximately 20 minutes of event data. With this threshold, both zooming and scrolling operate smoothly. Moreover, when the special attribute or the filtering attribute is modified, the changes take effect almost instantly, typically within a second. Notably, the tooltip display function and the single case connecting function remain unaffected by the number of events. Determining the upper limit of events that the methods can handle is challenging. At a setting of 20,000 events (which is equivalent to about 7 hours of event data in our examined processes) the methods still operate, but zooming and scrolling become noticeably sluggish. Additionally, any modifications to the special or filtering attributes may require approximately a minute for the visualizations to refresh.

It is important to note that the examined process has a brief lead time, ideally around 1-2 minutes, and operates with 8 parallel process executions. As a result, in different processes, the same number of events might span a more extended time period. Nonetheless, the primary objective of these visualization methods is to facilitate real-time process monitoring. Hence, the ability to closely examine just the most recent 20 minutes can be deemed adequate.

6. Conclusion

In this study, we have introduced two visualization methods. The first is tailored to the original process data, while the second is specifically designed for alignment data, with a particular focus on outputs from an alignment-based MOCC technique, specifically the multi-perspective prefix-alignment moves. Both of these methods position a user-selected attribute on the y-axis and timestamps on the x-axis while ensuring that visualized objects, representing event or alignment data, are displayed without overlap. When tested on real-life manufacturing data, our approaches effectively identified timing anomalies, mainly associated with machine downtimes, as well as consistent deviations that affected product quality. These findings emphasize the need for improved process monitoring and prompt corrective actions.

In our future work, we plan to enhance these visualization methods by boosting their performance and incorporating more interactive features for users. It would be advantageous to enable offline usage on a more extensive set of historical event data. Moreover, expanding the colouring options beyond just the activity attribute could be useful. For example, colouring objects based on the outcome of the process instances might offer additional insights to users.

7. Acknowledgements

Supported by the ÚNKP-22-3 New National Excellence Program of the Ministry for Culture and Innovation from the source of the National Research, Development and Innovation Fund.

8. References

[1] W. van der Aalst, Process Mining: Data science in action, 2nd. ed., Springer, Berlin, Heidelberg, 2016. doi:10.1007/978-3-662-49851-4.

[2] J. Carmona, B. van Dongen, A. Solti, M. Weidlich, Conformance checking: Relating Processes and Model, 1st. ed., Springer, Cham, 2018. doi:10.1007/978-3-319-99414-7.

[3] Z. Nagy, A. Werner-Stark, An alignment-based multi-perspective online conformance checking technique, Acta Polytech. Hung 19 (2022), 105-127.

[4] J. Carmona, B. van Dongen, M. Weidlich. Conformance checking: foundations, milestones and challenges, in: W. van der Aalst, J. Carmona (Eds.), Process Mining Handbook, 1st. ed., Springer, Cham, 2022, pp. 155-190. doi:10.1007/978-3-031-08848-3_5.

[5] N. Assy, C. Souchet, T. Bouffard, O. Anesini, Visualization Libraries for Process Analytics, in: M. Hassani, A. Koschmider, M. Comuzzi, F. M. Maggi, L. Pufahl (Eds.), Proceedings of the ICPM Doctoral Consortium and Demo Track 2022, volume 3299 of CEUR Workshop Proceedings, CEUR-WS.org, Aachen Germany, 2022, pp. 80-84.

[6] U. Celik, E. Akçetin, Process mining tools comparison. AJIT-e 9.34 (2018), 97-104. doi:10.5824/1309-1581.2018.4.007.x.

[7] P. Drakoulogkonas, D. Apostolou, On the selection of process mining tools, Electronics 10.4 (2021), 451. doi:10.3390/electronics10040451.

[8] D. Viner, M. Stierle, M. Matzner, A process mining software comparison, arXiv preprint arXiv:2007.14038 (2020). doi:10.48550/arXiv.2007.14038.

[9] A. Berti, S. van Zelst, D. Schuster, PM4Py: A process mining library for Python, Software Impacts 17 (2023), 100556. doi:0.1016/j.simpa.2023.100556.

[10] A. Kaouni, G. Theodoropoulou, A. Bousdekis, A. Voulodimos, G. Miaoulis, Visual Analytics in Process Mining for Supporting Business Process Improvement, in: C. Frasson, K. Kabassi, A. Voulodimos (Eds.), Proceedings of the 1st International Conference on Novelties in Intelligent Digital Systems, volume 338 of Frontiers in Artificial Intelligence and Applications, IOS Press, Amsterdam Netherlands, 2021, pp. 166-175.

[11] A. Berti, C. Y. Li, D. Schuster, S. J. van Zelst, The process mining toolkit (PMTK): Enabling advanced process mining in an integrated fashion, in: M. Jans, G. Janssenswillen, A. Kalenkova, F. M. Maggi (Eds.), Proceedings of the ICPM Doctoral Consortium and Demo Track 2021, volume 3098 of CEUR Workshop Proceedings, CEUR-WS.org, Aachen Germany, 2021, pp. 43-44.

[12] M. Song, W. van der Aalst, 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.

[13] I. Merkoureas, A. Kaouni, G. Theodoropoulou, A. Bousdekis, A. Voulodimos, G. Miaoulis, Smyrida: A web application for process mining and interactive visualization, SoftwareX 22 (2023), 101327. doi:10.1016/j.softx.2023.101327.

[14] D. Schuster, S. J. van Zelst, W. van der Aalst, Cortado: A dedicated process mining tool for interactive process discovery, SoftwareX 22 (2023), 101373. doi:10.1016/j.softx.2023.101373.

[15] A. Yeshchenko, J. Mendling, A survey of approaches for event sequence analysis and visualization using the esevis framework, arXiv preprint arXiv:2202.07941 (2022). doi:10.48550/arXiv.2202.07941.

[16] P. T. Hornix, Performance analysis of business processes through process mining, Master's thesis, Eindhoven University of Technology, Eindhoven, Netherlands, 2007.

[17] Z. Nagy, A. Werner-Stark, T. Dulai, Using process mining in real-time to reduce the number of faulty products, in: T. Welzer, J. Eder, V. Podgorelec, A. K. Latifić (Eds.), Advances in Databases and Information Systems: 23rd European Conference, ADBIS 2019, Bled, Slovenia, September 8–11, 2019, Proceedings, volume 11695 of Lecture Notes in Computer Science, Springer, Cham, 2019, pp. 89-104. doi:10.1007/978-3-030-28730-6_6.

[18] J. R. Rehse, L. Pufahl, M. Grohs, L. M. Klein, Process mining meets visual analytics: the case of conformance checking, arXiv preprint arXiv:2209.09712 (2022). doi:10.48550/arXiv.2209.09712.

[19] A. Adriansyah, B. F. van Dongen, W. M. van der Aalst, Conformance checking using cost-based fitness analysis, in: 2011 IEEE 15th International Enterprise Distributed Object Computing Conference, IEEE, 2011, pp. 55-64. doi:10.1109/EDOC.2011.12.
[20] F. Mannhardt, M. De Leoni, H. A. Reijers, The multi-perspective process explorer, in: F. Daniel, S. Zugal (Eds.), Proceedings of the Demo Session of the 13th International Conference on Business Process Management, volume 1418 of CEUR Workshop Proceedings, CEUR-WS.org, Aachen Germany, 2015, pp. 130-134.
[21] B. F. Hompes, J. C. Buijs, W. van der Aalst, A generic framework for context-aware process performance analysis, in: C. Debruyne, H. Panetto, R. Meersman, T. Dillon, E. Kühn, D. O'Sullivan, C. A. Ardagna (Eds.) On the Move to Meaningful Internet Systems: OTM 2016 Conferences: Confederated International Conferences: CoopIS, C&TC, and ODBASE 2016, Rhodes, Greece, October 24-28, 2016, Proceedings, volume 10033 of Lecture Notes in Computer Science, Springer, Cham, 2016, pp. 300-317. doi:10.1007/978-3-319-48472-3_17.
[22] T. Munzner, Visualization analysis and design, 1st. ed., CRC press, Taylor & Francis Group, New York, 2014.
[23] Y. Han, A. Rozga, N. Dimitrova, G. D. Abowd, J. Stasko, Visual analysis of proximal temporal relationships of social and communicative behaviors, Computer Graphics Forum 34.3 (2015), 51-60. doi:10.1111/cgf.12617.