Hindawi
Complexity
Volume 2022, Article ID 8670154, 13 pages
https://doi.org/10.1155/2022/8670154

Research Article

Simultaneous Process Mining of Process Events and Operator Actions for Alarm Management

László Bántay[1], **Gyula Dörgö**[1], **Ferenc Tandari**[2], and **János Abonyi**[1]

[1]*ELKH-PE Complex Systems Monitoring Research Group, University of Pannonia, Veszprém H-8200, Hungary*
[2]*MOL Danube Refinery, Százhalombatta H-2443, Hungary*

Correspondence should be addressed to János Abonyi; janos@abonyilab.com

Received 1 June 2022; Accepted 1 August 2022; Published 19 September 2022

Academic Editor: Andrea Murari

Alarm management is an important task to ensure the safety of industrial process technologies. A well-designed alarm system can reduce the workload of operators parallel with the support of the production, which is in line with the approach of Industry 5.0. Using Process Mining tools to explore the operator-related event scenarios requires a goal-oriented log file format that contains the start and the end of the alarms along with the triggered operator actions. The key contribution of the work is that a method is presented that transforms the historical event data of control systems into goal-oriented log files used as inputs of process mining algorithms. The applicability of the proposed process mining-based method is presented concerning the analysis of a hydrofluoric acid alkylation plant. The detailed application examples illustrate how the extracted process models can be interpreted and utilized. The results confirm that applying the tools of process mining in alarm management requires a goal-oriented log-file design.

1. Introduction

The motivation of the present work is to develop a methodology for the process mining-based analysis of alarm and event-log databases to increase process safety and reduce the workload of the operators. As a result, we will be able to understand the chain of events that trigger an operator action, as well as explore the effects of different operator action strategies; from another point of view, to gain the models of processes, leading potentially to malfunctions or safety incidents. With the help of this knowledge, a better designed and more effective alarm management [1] and operator training system can be developed.

The Industry 5.0 approach considers the wellbeing of the workers in productivity and efficiency improvement projects more. State-of-the-art industrial production systems contain complex process control solutions. The amount of recorded signals and process variables makes it difficult to have a clear view of the relationships between the different process elements; the control of the processes can be a demanding task for the workers. To lower the workload of the operators, a good understanding of the process element relationships is needed to predict the probable event scenarios that can be the basis of a decision-supporting system. The work of the operators can be reduced and supported in other ways as well. Generally speaking, predictable alarms do not contain useful information for the operators. Therefore, automation solutions should handle them before they occur, or completely useless alarms should be suppressed before an announcement is made [2]. Future alarm sequences can also be predicted using historical knowledge of the process [3]. The exploration of operational strategies holds promising opportunities for automation as well: the sequences of alarms and the corresponding operator actions can be determined besides the best operational practices [4]. The analysis of the announced alarms can also facilitate root cause analysis [5]. A wide range of data-based solutions have been applied to reduce the operator workload, e.g., the conventional techniques of deadbands [6], delay timers [7], or filtering [8]. Advanced alarm management solutions aim to define more informative features for the operators by identifying redundant and co-occurring alarms. Two main approaches are known. First, correlation analysis-based techniques are widespread, where the aim is to find frequently co-occurring alarms over a short period of time, which can be considered redundant [9]. Second, the frequently occurring longer operational patterns can be considered to be the symptoms of the same malfunction and can be revealed by frequent pattern mining algorithms [10], as well as applied in terms of alarm prediction and suppression [2]. It is also advantageous to apply highly efficient data-driven solutions, like deep learning [11] or decision tree-based classifiers [12]. To gain understandable models that can be directly used in alarm management is challenging. To explore complex event chains or comparable models of operator action strategies, determining the correlation of event pairs, deep learning methods with hard-to-understand results are not satisfying. For compact and comprehensible models, we need to apply process mining.

Process mining is a collection of techniques that support the understanding of processes based on event logs [13]. Process mining algorithms are applied in various fields, without the aim of completeness: in healthcare to improve the operational efficiency of processes [14], in business management to analyze the processes and reveal their bottlenecks [15], for the automatized analysis of financial statements during audit processes [16], or the support of e-learning are among the applications as well [17]. Moreover, a recent and highly promising application field is the identification of repetitive processes using process mining to discover potential processes for robotic process automation [18]. For a recent collection of research fields and application areas in process mining, please refer to the work of Garcia et al. [19]. These process mining algorithms are tailored to goals for the purpose of discovering process flows, where the events of the different processes are organized into traces. A trace is a collection of events that are considered to be related in some way. The definition of the traces is essential for the purpose of process discovery, especially in the case of alarm management, where the connection between alarms and operator actions requires accurate trace generation. Therefore, the structures of alarm and event logs are unsatisfactory in terms of process mining as they lack this very important component, that is, trace indicators. A process mining-based approach was applied to evaluate the behavior of plants and support the rationalization of alarms. Its basic concepts are presented in Reference [20], while the applicability of process mining algorithms for the determination of alarm performance metrics and the exploration of process behavior are presented in Reference [21], which can be considered to be the motivation of the present study.

Although these studies usually focus on different aspects of alarm management, e.g., operator actions [4] or alarm rationalization [21], they lack the general formalization of the problem and the goal-oriented definition of the input database for different purposes of analysis. Given the lack of a comprehensive study on the applicability of the techniques of process mining with regard to large-scale industrial alarm and event log databases, the contribution of the present work is the following:

(i) We discuss the goal-oriented tasks and the related definition of the events, as well as further characteristics of the events and resources. Traces must be defined to provide a suitable structure of alarms and the related operator actions, resulting in the most appropriate input dataset for the applied mining algorithms.

(ii) The effectiveness and applicability of the proposed methodology are presented with regard to the analysis of the large-scale alarm and event-log database of an industrial plant. We have gained understandable and comprehensive information that is useful not only in alarm management and operator training systems but also in the process mining of other industrial sectors.

According to the contributions, the core applicability of the proposed methodology is not narrowed down to the industrial alarm systems but can be transferred to other fields of applications as well, where temporal events are present and their follow-up or cause and effect type of relationship is to be analyzed (similarly to the back and forth relationship of alarms and operator actions). In the case of alarm systems, the type of process control system, let it be a distributed control system (DCS), supervisory control and data acquisition system (SCADA), or any other type of system, is irrelevant, as long as the provided alarm (or event) data are timely and accurate. Using the proposed methodology, the alarm evolution paths can be identified, the triggering alarms of operator actions can be revealed, and recommendations for the reduction of the probability of hazardous situations can be provided.

The structure of the work is the following. In Section Materials and Methods, we provide a brief overview of industrial log files (structure and content), introduce the goal-oriented definition of the traces, identify the necessary rules to generate traces, and discuss the process mining tools necessary to achieve our goals. In Section Results and Discussion, we examine the previously defined process mining tasks, the preparation of the log file, the time distribution analysis of the events (alarms and operator actions), the alarm spillover analysis between the production units, and the discovery of the connection between alarms and operator actions. In Section Conclusion, some concluding remarks are provided, experiences are discussed, and possible future research directions are identified.

2. Materials and Methods

This section introduces the basis of process discovery (log files), the event-clustering method (trace generation rules), and the goal-oriented selection of process exploration tools. As in the case of any project-like activity, an execution plan to achieve our predefined goals must be drawn up. In terms of the present case study, a schematic summary is provided in Figure 1.

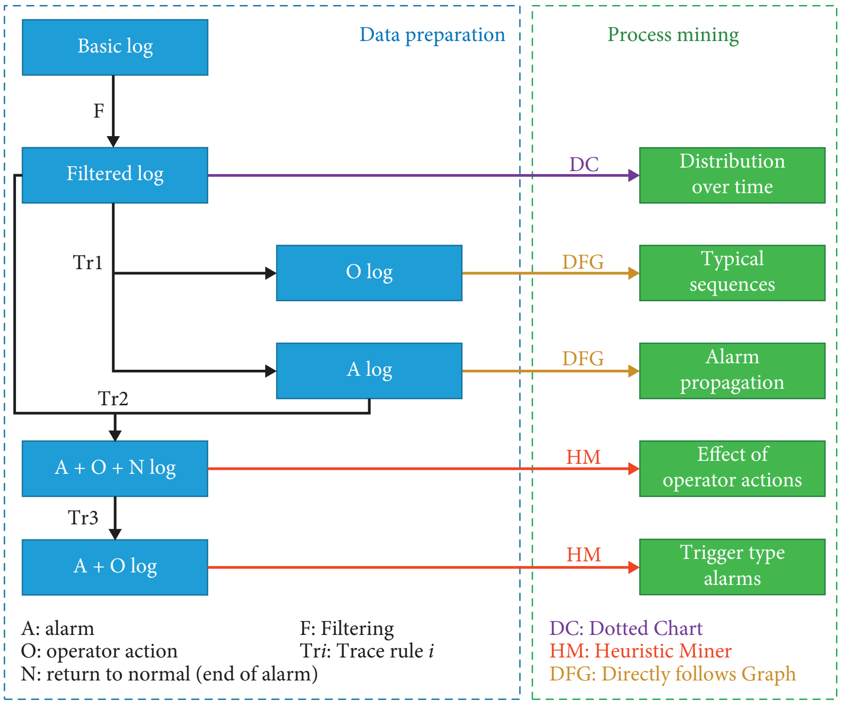


FIGURE 1: The concept of process mining-based alarm management. After preparing the data, we can perform our process discovery tasks on our sub-logs that were generated by applying the suggested trace rules.

2.1. The Structure of the Alarm and Event Log Databases. Discrete events, e.g., alarms and warnings, are recorded in the process control unit of almost every production site when a variable exceeds its associated threshold. An industrial alarm and event-log database is usually composed of these alarms and warnings, operator actions, system messages, as well as any further timestamped information logged by the control system. For the purpose of further mathematical formulation, every event can be regarded as a state of the technology (denoted by s) represented by $<\mathrm{pv}, a>$ data pairs. pv indicates the name of the process variable, while a represents the attribute that indicates the state occurring on the given variable, e.g., high or low alarm in the case of an alarm announcement or increase or decrease (open or close) in the case of operator action. An *event* is the occurrence of a given state, which can be an event of temporal duration such as alarm messages or point-like in time such as operator actions. For example, the description of an alarm and an operator action can be mathematically represented as $<$ column inlet temperature, high alarm $>$ and $<$ cooling water inlet valve, open $>$, respectively. The events are represented by $< s, st, et >$ triplets, where s denotes the state that occurs between st starting and et ending times. For an event with a point-like temporal characteristic, even though the st and et times are regarded as equal to maintain a uniform mathematical formulation, it must be decided whether to keep both or utilize only the st starting time for further processing. This question is addressed later when the XES file is considered.

The files of industrial alarm and event logs are usually composed of timestamped events of alarms, operator actions, system messages, and any further temporal information. Additional information can help us to determine which sensor generates the alarm in the process and the priority of that alarm, e.g., the location of the event, that is, in which part of the plant/organization it occurred. Each event that occurs is labeled with a tag name. Every alarm and operator action can be considered to be the state of the process. The different alarms in the event log have starting and end times. The starting time is when the alarm in the process was raised, and the end time is when the process returned to its proper operational zone. The operator actions are usually considered to be point-like events, i.e., their starting and end times coincide. A series of events can be defined as a trace. Let L denote a set of events; $\sigma \in L$ stands for an event trace, that is, a sequence of events; $T \subseteq L$ represents an event log, i.e., a set of event traces. Besides information on the occurrence of an event, the log files usually contain other information as previously discussed, e.g., the location of the sensor that raised the alarm in the process, which is possibly categorized into units or production units, etc. This work demonstrates the importance of task-specific trace definitions. The methodology of trace segmentation will be discussed later.

Our goal is to identify basic patterns in the chain of alarms to focus on frequent sequences that can help us compile a prediction model of the alarms. In Table 1, an example of an industrial alarm and event-log can be seen. The column labeled "Tag" has been added to support the analysis and is a summarized representation of the sensor's name. The first part is the name of the tag, the second and third are those of the unit and production unit, respectively, while the last one is the type of event (A: alarm, O: operator action, N: return to normal). Different "sub-logs" are

Table 1: An example of an industrial alarm and event log file.

ID	Tag name	Event	Start time	End time	Unit	Prod. unit	Tag
1	321	A	2018.05.01 00:01:01	2018.05.01 00:03:08	3	1	321_3_1_A
2	632	A	2018.05.01 00:01:03	2018.05.01 00:10:06	4	5	632_4_5_A
3	421	A	2018.05.01 00:01:10	2018.05.01 00:05:10	2	5	421_2_5_A
4	312	A	2018.05.01 00:01:30	2018.05.01 00:03:08	3	1	312_3_1_A
5	321	O	2018.05.01 00:02:10	2018.05.01 00:02:10	3	1	321_3_1_O
7	321	N	2018.05.01 00:03:08	2018.05.01 00:03:08	3	1	321_3_1_N
8	421	O	2018.05.01 00:04:01	2018.05.01 00:04:01	2	5	421_2_5_O
10	632	N	2018.05.01 00:10:06	2018.05.01 00:10:06	4	5	632_4_5_N

Table 2: Task oriented event types in sub-logs and suggested tools to process (A: alarm, O: operator action, N: return to normal, PM: process mining).

Object	Task	Event types	Tool
Basic log	Data preparation for PM	A, N, O	Filtering and XES generator
Alarms	Distribution over time	A	Dotted chart
Alarms	Typical processes	A	Directly-follows graph
Alarms	Spillover among units	A	Directly-follows graph/Heuristic miner
Alarms	Trigger type events	A, O	Heuristic miner
Operator actions	Distribution over time	O	Dotted chart
Operator actions	Typical processes	O	Directly-follows graph
Operator actions	Effect of operator actions	A, N, O	Heuristic miner

necessary to examine various mining tasks. The required types of events are summarized in Table 2. The object of the analysis indicates what information is of interest with regard to the analysis provided in the task column. The event types indicate the types of events available for analysis. Finally, the suggested tools to process are mentioned. The three types of objects are the following:

(i) Basic log: all three types of events are needed, after a filtering/cleaning step, the log file has to be put into a standardized format. This format is XES (Section 2.2).

(ii) Alarms: dotted chart, directly-follows graph, and heuristic miner are proper tools to analyze the different aspects of alarm propagation. By adding operator actions to the traces, their triggering alarms can be identified.

(iii) Operator actions: the tools and tasks are more or less the same as in the case of alarms. The most complex task is to explore the effect of the operator actions, this needs all three types of events, as alarms are the trigger events and return to normal events are the consequences of operator actions.

There are two kinds of tools to process the data, the preparation type and the ones responsible for the Process Mining part itself. The filtering/cleaning step is a standard data processing task; it can be tailored and automatized by using the Python programming language, as well as its transformation to XES standard. The three Process Mining tools are also available in Python. This way, it is quite easy to develop an integrated solution tailored for the actual purpose.

2.1.1. Dotted Chart Analysis. The most transparent method by which to visualize the event log is the dotted chart analysis. In these charts, a dot represents a single event in the log with two orthogonal dimensions, namely, time and component types. Component types like instance, originator, task, event type, or data elements are shown on the vertical axis. Time is measured on the horizontal axis of the chart. Many measures related to events can be determined, such as the average number of events occurring over a certain time period, the maximum number of events in that time period, the maximum and minimum time interval between events, etc. Time can be presented factually or relatively. The relative time can be used to abstract the log file. For every component type, the first event is positioned at time 0 and subsequent events are placed relative to the time of occurrence of the first event. Moreover, the shape and color of the dots can be changed depending on the examined event attributes, adding dimensions to our chart.

From a chart like this, a lot of useful information can be obtained, e.g., where the alarms occur more frequently or which production units are affected more by alarm events.

2.1.2. Directly-Follows Graph. On a directly-follows graph, an edge is represented between two nodes when at least one trace where the target event follows the source event is present. In a nutshell, this method by which a DFG is obtained ([22]):

(i) defines 3 parameters, namely, τ_{Var}, τ_{act}, τ_{df}

(ii) removes cases with a frequency lower than τ_{Var} from the log

(iii) removes events with a frequency lower than τ_{act} from the filtered log and adds a node for each of the remaining activities

(iv) connects the nodes where 2 activities follow on from each other at least τ_{df} times

On these graphs, two metrics can be represented, namely, frequency (the number of times the target event follows on from the source event) and performance (the average time elapsed between the source and target events), the decision depends on the type of required information.

2.1.3. Heuristic Miner Algorithm. The heuristic miner algorithm explores the control-flow perspective of the process model. The log is analyzed for the presence of causal dependencies. If an event is always followed by another event, a dependency relationship probably exists between these events. The log should be analyzed for these causal dependencies. The advantage compared to α-miner is that the heuristic miner algorithm considers frequencies and can handle skipping activities [13]. Several parameters can be adjusted, e.g., minimum activity count. Events that occur under this threshold are not shown on the nets, which can be a Heuristic net or a Petri net. Another parameter is the minimum DFG occurrences, which is the minimum number of occurrences of an edge to be considered. This attribute shows that the heuristic miner is based on DFG. Heuristic mining requires a clear starting and end event, assuming that every activity is located on a path from the starting activity to the end activity. As is the case in DFG, frequency and performance parameters can be entered on the net.

2.2. Goal-Oriented Definition of the Traces. The input of process mining tools is a log file in a standard format; in this work, we have chosen the XES standard. XES is an XML-based standard for event logs. Its purpose is to provide a generally acknowledged format for exchanging event log data between tools, applications, and domains. The main reason for choosing the XES model is the support it provides for traces. As mentioned above, it is very important that traces are well defined. Usually, in industrial log files, the events are sequenced independently of one another, unlike in the XES standard. The majority of process mining algorithms take into consideration traces in addition to events.

For the effective mining of the alarm and event-log files, a definition of the time window applied for the segmentation of the alarm and event log files into traces that is dependent on the purpose is required.

As previously mentioned, since traces play a significant role in process mining methods, the task-dependent determination of trace-defining time windows is required. As the main event of the alarm log file is the appearance of the alarm event itself (that is, its starting time), the basis for the trace generation is the following:

Let α and β denote two consecutive events in log L. Let $T(\alpha)$ and $T(\beta)$ stand for the times of occurrence of events α and β, respectively, and σ represents the spillover constant. If $T(\beta) - T(\alpha) < \sigma$, then α and β are located on the same trace. However, if $T(\beta) - T(\alpha) \geq \sigma$, then β is placed in the following trace. The spillover constant can be tuned based on the knowledge about the dynamics of the system. Obviously, it can be calculated with the help of the experience of the operators and identification of data driven dynamical models (Figure 2).

There are three areas to discuss the rules concerning trace generation, the analysis of alarms, operator actions, and their relations.

2.2.1. Analysis of Alarms. The first concept to explore in an alarm management system is the spillover effect of the alarms. These sequences of alarms are caused primarily by the decline in product streams, as well as the spread of pressure anomalies or attributes (temperature and concentration) connected to technology streams. According to this, probable propagation times are related to the sojourn times of equipment, the length of pipelines, and the logic of the control system. Hazard and operability analysis (HAZOP) provides options to explore this malfunction propagation, in addition, more and more attention is being paid to dynamic HAZOP. As its automated use is challenging [23] and these methods are very resource-demanding (require expert engineers), it would be beneficial to explore these potential relationships automatically from the log files. In this case, it is practical to use a rule of thumb to define the possible propagation times. This rule can be determined by analyzing the time of occurrence of consecutive events originating from different production units.

2.2.2. Analysis of Operator Actions. A similar analysis can also be performed on operator actions. Despite being a complex troubleshooting process that can last for hours, our primary goal is to identify the sequences of correlated operator actions (similar to parent-child type alarms). One way of achieving this is to define a time window lasting between 10 and 60 seconds (based on the cognitive model of operators and the attributes of the existing system). A new series of actions is identified if the time gap between two consecutive actions exceeds this time window. Another way is to regard alarm acknowledgements as the end of intervention activities (if this type of event is found in our log file). This way, they generate the groups of action series.

2.2.3. Connection between Alarms and Operator Actions. The most complex task is to analyze operator actions with regard to the alarms that trigger them and qualify the efficacy of the interventions.

To determine which operator actions trigger alarm events, operator actions should be placed into our existing alarm traces (which commence after the starting time of the trace and finish before the end time of the trace or the starting time of the following trace).

If the aim is to explore the effect of the operator actions, the end times of the alarms should be put into the aforementioned traces, as the Return to Normal pair of alarm events.

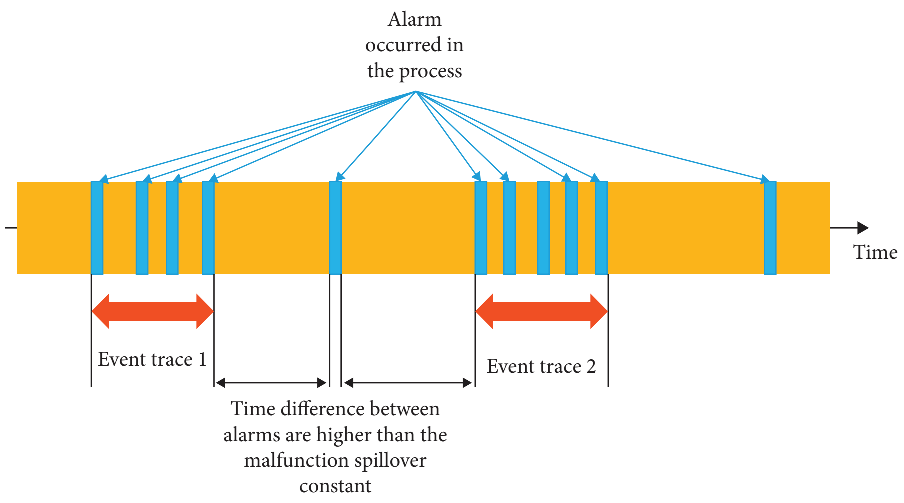

Alarm occurred in the process
Time
Event trace 1
Event trace 2
Time difference between alarms are higher than the malfunction spillover constant

FIGURE 2: The segmentation of an event log database into event traces of potential propagation of error.

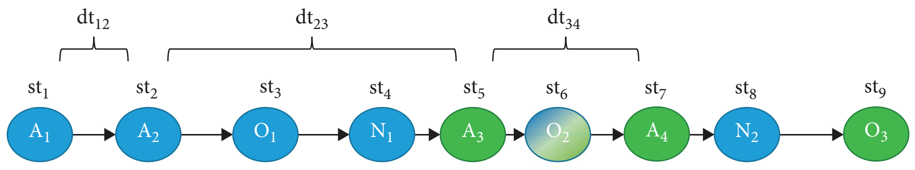


FIGURE 3: st_i denotes the start time of the event, dt_{ij} indicates the time difference between alarm start times, and dt_w indicates the time window and T_i the trace. If $dt_{12}, dt_{34} < dt_w$ and $dt_{23} > dt_w$ and $\forall st$: $st_i < st_{i+1}$, then $A_1, A_2, O_1, N_1, O_2, N_2 \in T_1$ and $A_3, O_2, A_4, O_3 \in T_2$. It is worth to note that operator action O_2 belongs to two traces.

(1) Rules of Generating Traces. In the previous list, trace generating methods were identified; now, the formalization of these is provided. At the start, we consider one trace, which contains all events, and with the following methods we will split it step by step.

$A_{j,n}$, $O_{k,n}$, and $N_{p,n}$ are the jth alarm, kth operator action, and pth return to normal event of the nth trace (σ_n), respectively, where $n \in \{1, \ldots, |\sigma_i|\}$, $j \in \{1, \ldots, |A_i|\}$, k and $p \in \{1, \ldots, |N_i|\}$. tw_1 is the time window constant for alarms, tw_2 is the time window constant for operator actions, and $t(A_{j,n})$, $t(O_{k,n})$ and $t(T_{n,n})$ are the timestamps of the related events. The explanation of the rules and their mathematical description is as follows.

Trace rule 1: $t(A_{j+1,n}) - t(A_{j,n}) > tw_1, A_{j+1,n} \longrightarrow A_{1,n+1}, A_{j,n} \longrightarrow A_{|A_i|,n}$: if the difference between the timestamps of two consecutive alarms is greater than tw_1, then the alarm with the higher index will be the first event of the next trace and the one with the lower index will be the last event of the actual trace. This rule can be applied to operator actions as well. These traces provide the input to gain the distribution of events over time, the identification of typical event sequences and the spillover of the alarms among production units.

Trace rule 2: From traces made by Trace rule 1, we generate the return to normal events from the end timestamps of the alarms. This means that the number of return to normal events will be equal to the number of alarm events $(|A_i| = |N_i|)$ and traces will lap over each other, as an alarm of a trace can end later, than the start time of the next trace's first alarm. We put an operator action into the trace, if its timestamp is later than the first alarm of the trace and sooner than the last return to normal event of the trace $(t(O_{1,n}) > t(A_{1,n})$, $t(O_{|O_i|,n}) < t(N_{|N_i|,n}))$. A visualized example is shown in Figure 3. From these traces, the effect of operator actions can be gained. To identify trigger type alarms, return to normal events have to be removed from the traces made by Trace rule 2. Obviously, they should not be excluded, but process mining a log without unnecessary events results in a more clear process model.

2.3. Process Mining-Based Alarm Management Solutions. In this section, the theoretical background of the applied analysis methods is presented.

Different algorithms of process mining can help us to identify patterns within the swarm of data placed in the log files. Given the need to find a solution to this common problem, different process mining techniques and several software products to evaluate the data mining tasks can be used. In this work, instead of using the well-known and usual Process Mining tools (like ProM [24] or EMiT [25]), we have used an open-source Python programming language-based solution, PM4Py. This library provides a wide range of process mining tools and since it is based on Python, which is a great tool for manipulating huge data sources (like industrial log files), PM4Py is an excellent option to develop semi- or fully automated process discovery methods from

Table 3: Tasks and process mining tools (A: alarm, O: operator action).

Task	Tool
Distribution over time (A and O)	Dotted chart
Typical sequences (A and O)	Directly-follows graph
Spillover among units (A)	Directly-follows graph/Heuristic miner
Trigger type alarms	Heuristic miner
Effect of operator actions	Heuristic miner

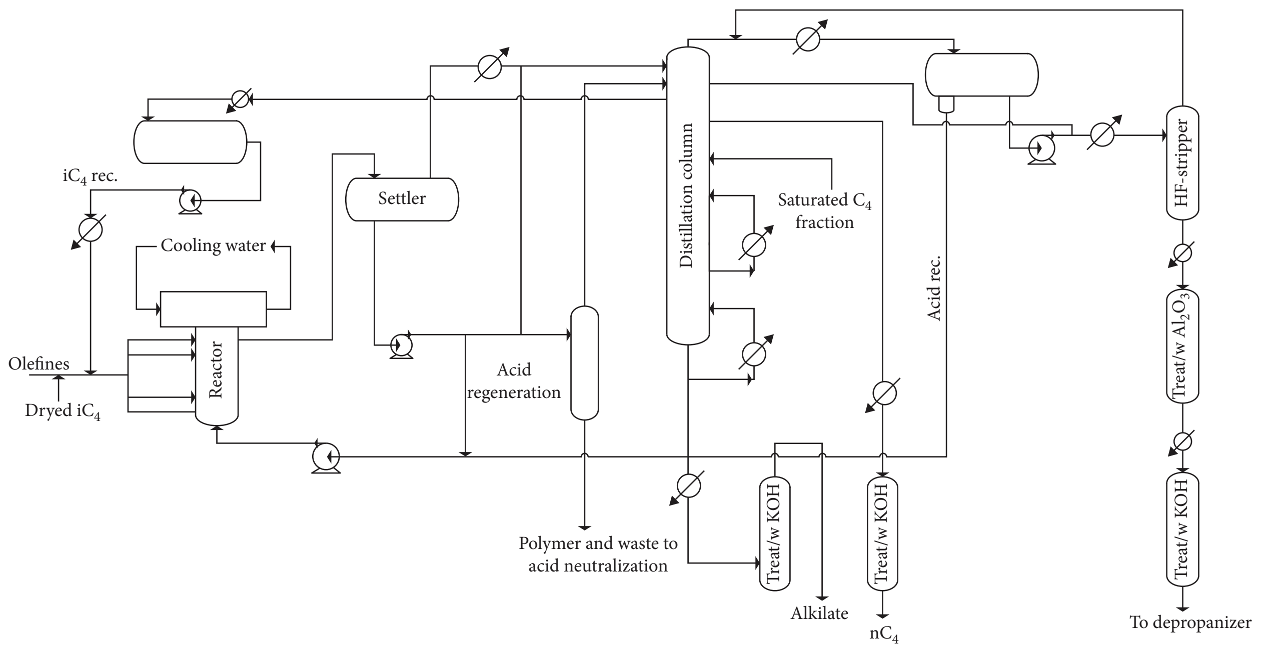


Figure 4: Schematic diagram of the plant in the case study.

scratch on one platform. The tasks and tools to be used are summarized in Table 3.

3. Results and Discussion

In order to show the industrial applicability of the formerly introduced process mining method, the alarm management system of an industrial hydrofluoric acid alkylation plant will be analyzed. The process flow diagram of the plant can be seen in Figure 4. The plant consists of four production units and more than 400 tags, the distributed control system is a Honeywell product. The logic of the tag names is *X*,. . .,*X_YY_Z*, where *X*. . .*X* is the identifier of the tag, *YY* is the identifier of the production unit (where the tag is located), and *Z* is the type of the event. The identifiers of the production units are the following: CH: isostripper, propane-depleting and propane handling unit; U1: utility streams; AC: reactor and acid generating unit; FD: raw material and drying unit; 02: "virtual" unit, collection of sensors, that cannot be assigned to a specific unit. There are three types of events, namely, alarm (denoted with A), operator action (denoted with O), and return to normal (denoted with N).

Even though an alarm rationalization was performed on the plant, so the events in the log files are all considered to be relevant by the operating personnel, the log files need to be processed carefully, as remaining problems can be present, which can cause issues while undertaking the process discovery tasks.

3.1. Preparation of the Log File of the Alarm System. The log file of the aforementioned plant contains a lot of data, in excess of 200,000 events over a time period of four months. As previously discussed, the log file must be filtered to ensure only relevant and valuable data for the purpose of process mining is retained. The minimum number of attributes for process mining is three: an identifier of the event, at least one timestamp of the event (start or complete), and an identifier of the trace. Although not mandatory, it is useful to have additional attributes, for example, the name of the resource that triggered the event, the name of the organizational group in which the resource is located, and in the case of temporal-type events, the counterpart of the timestamp (start or complete) and the type of the event.

Another aspect that must be taken under consideration is what type of events to keep. Ten event types are present in the analyzed log file, namely, alarm, return to normal, acknowledge, operator action, system, operator message, suppress, shelved, unshelved, process event. According to our goals, first, it was decided to retain three types of events, that is, alarm, return to normal, and operator action. Alarm

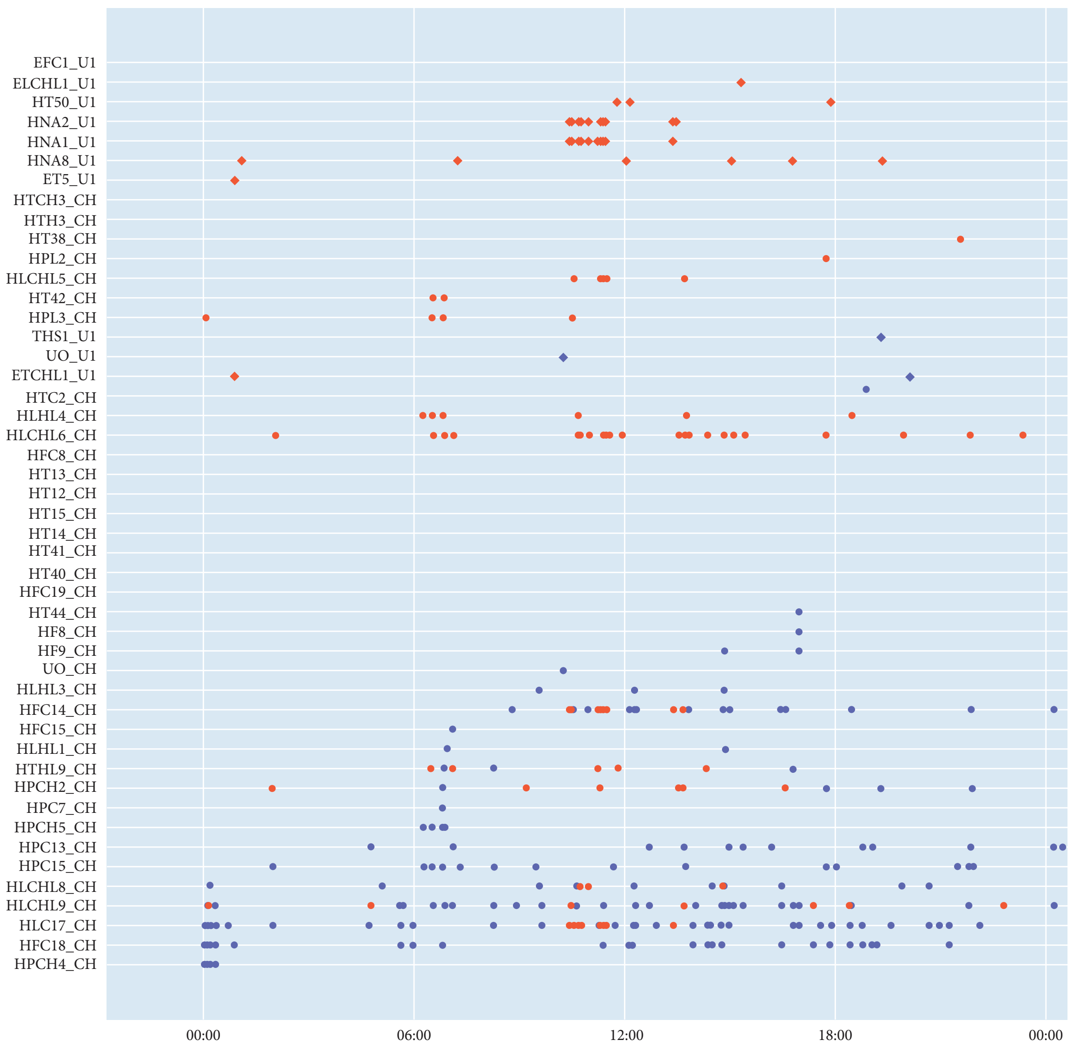


Figure 5: Distribution of alarms and operator actions over time (red: alarm, blue: operator action). The x axis shows the time and the y axis shows the name of the tags.

and operator action will be used to explore their number of occurrences and the sequences in which they occur; moreover, all three will be used to determine the causal relations between the alarms and operator actions. As a result of filtering out types of events, approximately 80,000 events remained.

As soon as the sub-logs with the needed types of events are obtained, traces must be generated. Trace window constants have to be defined for every sub-log, the object and the contained event types of these sub logs are collected in Table 2. The value of these constants depends on the actual system, literature data, and industrial experiences.

3.2. Distribution over Time and Typical Event Chains. To visualize the distribution of the events over time, a dotted chart is an excellent tool to use. Formerly, time distribution analysis has been identified as a task for both alarms and operator actions. As it can be interesting to compare the time of occurrence of alarms with the time of occurrence of operator actions, both have been visualized on one dotted chart. Figure 5 shows a one-day-long time window of production units CH and U1; the colors represent the types of events. It can be seen that although tags are present where both Alarm and OperatorAction events occur, many can be identified where this is not the case. It is not inevitable that interventions are made at the same place where the alarm occurs. For example, only alarms are located in production unit U1, in production unit CH, more operator actions are present than alarms. It can be supposed that the alarms in U1 trigger alarms in CH which in turn trigger the operator actions.

By briefly examining the time distribution statistics, it can be seen that either extremely long-lasting and near-to-zero second long alarms are present. These extreme values can have a biased effect on our process mining results. These events contain little or no timely information for the operators were filtered out from the log file. Only alarms that lasted between 5 and 28,800 seconds (8 hours) were retained. Of course, the upper and lower limits should be considered based on the given system and task.

Now that our log file has been “normalized,” the next step is to generate the traces. If the needed trace windows are to be determined, the statistics regarding the time difference

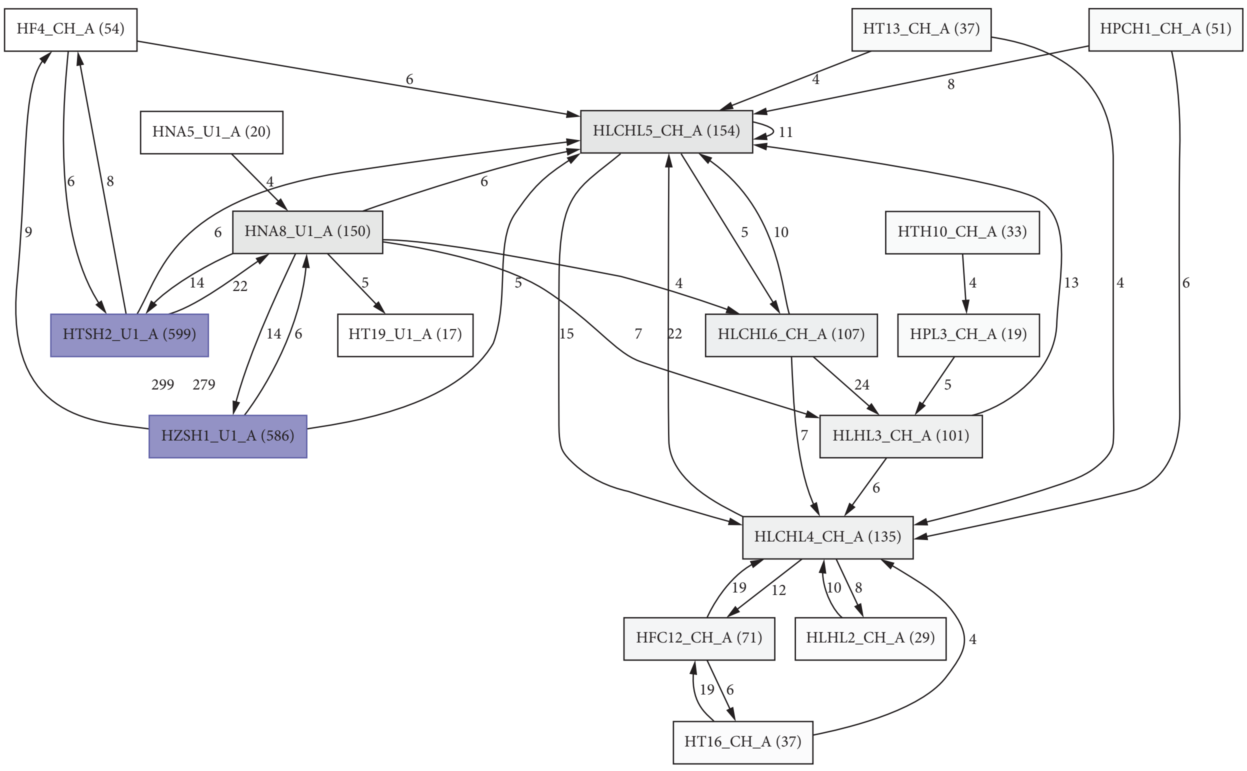


Figure 6: Alarm series (part of the DFG). CH: isostripper, propane-depleting, and propane handling unit; U1: utility streams.

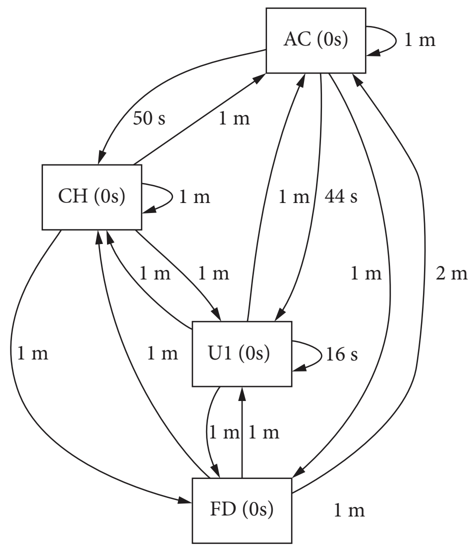


Figure 7: Alarm propagation times between units. CH: isostripper, propane-depleting, and propane handling unit; U1: utility streams; AC: reactor and acid generating unit; FD: raw material and drying unit.

between the starting times must be examined. According to the statistics concerning the alarm starting times, for exploring the typical alarm chains, the trace window was chosen to be 220 seconds (the median value). To ensure at least two events are found in a trace (which is the minimum to be considered in a chain), the traces consisting of one event must be removed.

The first tool that can be used to explore the typical alarm chains is the directly-follows graph (DFG). The frequency of both the events and nodes can be put on the graph. Figure 6 shows a portion of the DFG of the alarms, as the original graph is too large to be presented in full here.

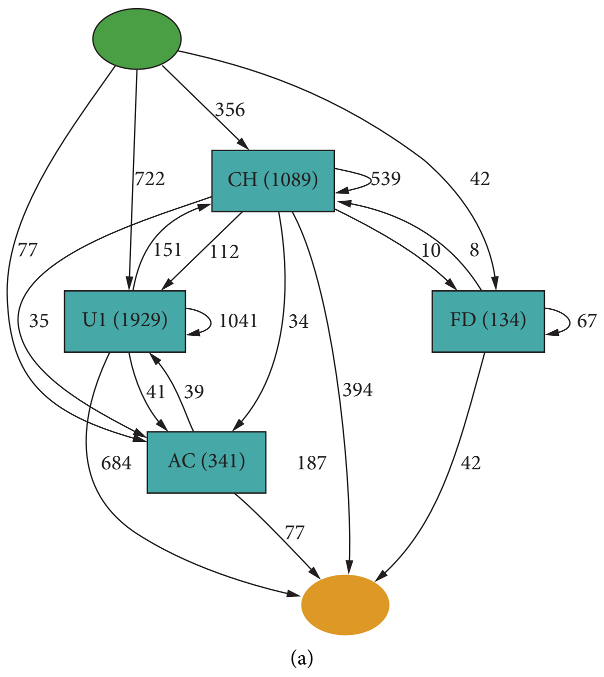
Suppose the typical alarm propagation time between the units is sought, a DFG can be used once again, where the average elapsed time between two alarms occurring in different units is added, as is presented in Figure 7:

Although the average times are more or less identical, the frequency at which the alarms occur in unit U1 is much higher than in the other units. Zero seconds can be seen in the boxes of the units because the events were regarded as point-like, so they are dimensionless in terms of time.

Even though a DFG can provide a good overview of typical sequences, a different tool must be used to explore processes. One option is the heuristic miner algorithm to obtain a so-called heuristic net, which is one form of visualizing the typical processes. As frequent event chains are to be explored, the parameter minimum activity count was used and set at 100 and 500, as presented in Figure 8. On a heuristic net, the green ellipsis represents the start and the orange one represents the end of the process.

Obviously, the biggest proportion of the alarm events occurs in unit U1, which has a high rate of interaction with units CH and AC. Although our assumption that alarms in U1 trigger alarms in CH is proven, the opposite can also occur. This shows the advantage of a heuristic net over a dotted chart or a DFG.

To explore the typical series of operator actions, the formerly presented DFG is used. Figure 9 shows the



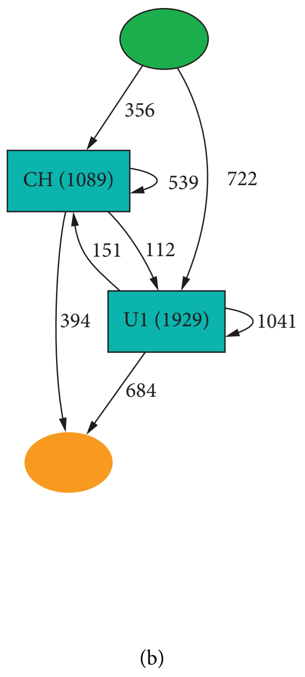


Figure 8: Heuristic nets of alarm propagation between production units. CH: isostripper, propane-depleting, and propane handling unit; U1: utility streams; AC: reactor and acid generating unit; FD: raw material and drying unit. (a) Minimum activity count = 100. (b) Minimum activity count = 500.

frequency of two consecutive operator actions. It can be seen that the most operator actions are located in unit CH, as was presumed from Figure 5. Nodes denoted in a darker color with thicker edges represent more frequent events and transitions. From this graph, the group of tags where the most of the operator actions occur can be identified, along with information about the frequent series of operator actions.

3.3. Correlation between Alarms and Operator Actions.

3.3. Correlation between Alarms and Operator Actions. To understand the connection between operator actions and alarms, two different questions may have to be answered:

(1) Which alarm triggers an operator action?

(2) What is the effect of the operator actions?

To answer these two questions, different log file and trace generation rules are required. The first requires the event types alarm and operator action, while the second requires return to normal events, which can be a result of an operator action. The trace generation rules were determined in Section 2.2. First, our formerly generated alarm traces are taken and return to normal events are generated from the end timestamps of the alarms. Subsequently, operator actions are placed into our traces with timestamps between the first and last events in the trace (practically speaking, the first alarm and the last return to normal events). By considering this method, trigger-type alarms and the effect of operator actions can be handled in one task as the log for trigger-type alarms would also be the one for exploring the effect of operator actions in the absence of the return to normal events.

Figure 10 shows the explored processes (minimum DFG was set at 55). By closely examining the net, two main types of processes (in addition to a third one) can be identified. For the purpose of better readability, starting and end points of the alarm sequences have been highlighted with colored boxes (same color for each pair). The boxes with dashed lines denote those processes where no operator action occurred between the starting and the end of an alarm sequence (process type 1). The ones denoted with solid lines mark processes containing operator action (process type 2). The third type, denoted by dotted lines belongs to both, as this alarm (HNA8_U1_A) can end either with or without an operator action (process type 3). Furthermore, an area which is worth examining closely is the green ellipse, as this part definitely appears to be a typical process (this "group" can also be seen in Figure 9). From this net, which tag is relevant in which kind of process discovery task can be determined, moreover, our log file can be filtered further and the mining conducted again.

The HLCHL9_CH_A alarm indicates problems with the liquid level at the HF stripper bottom. It is well visible that the operators frequently apply the HFC17_CH_O action in this situation, which modifies the bottom inlet of the HF stripper. Similarly, they apply the HFC18_CH_O action in this situation, which controls the steam inlet of the re-boiler of the stripper. In the case of the events marked by black brackets, the HPCH2_CH_A alarm indicates problems with the depropanizer pressure, while the action applied in this situation, HFC15_CH_O, controls the blow off of the technology. The HTSH5_U1_A and HZSHP1_U1_A alarms (dark blue and brown dashed brackets) almost always co-occur. As these alarms both related to the problem of the

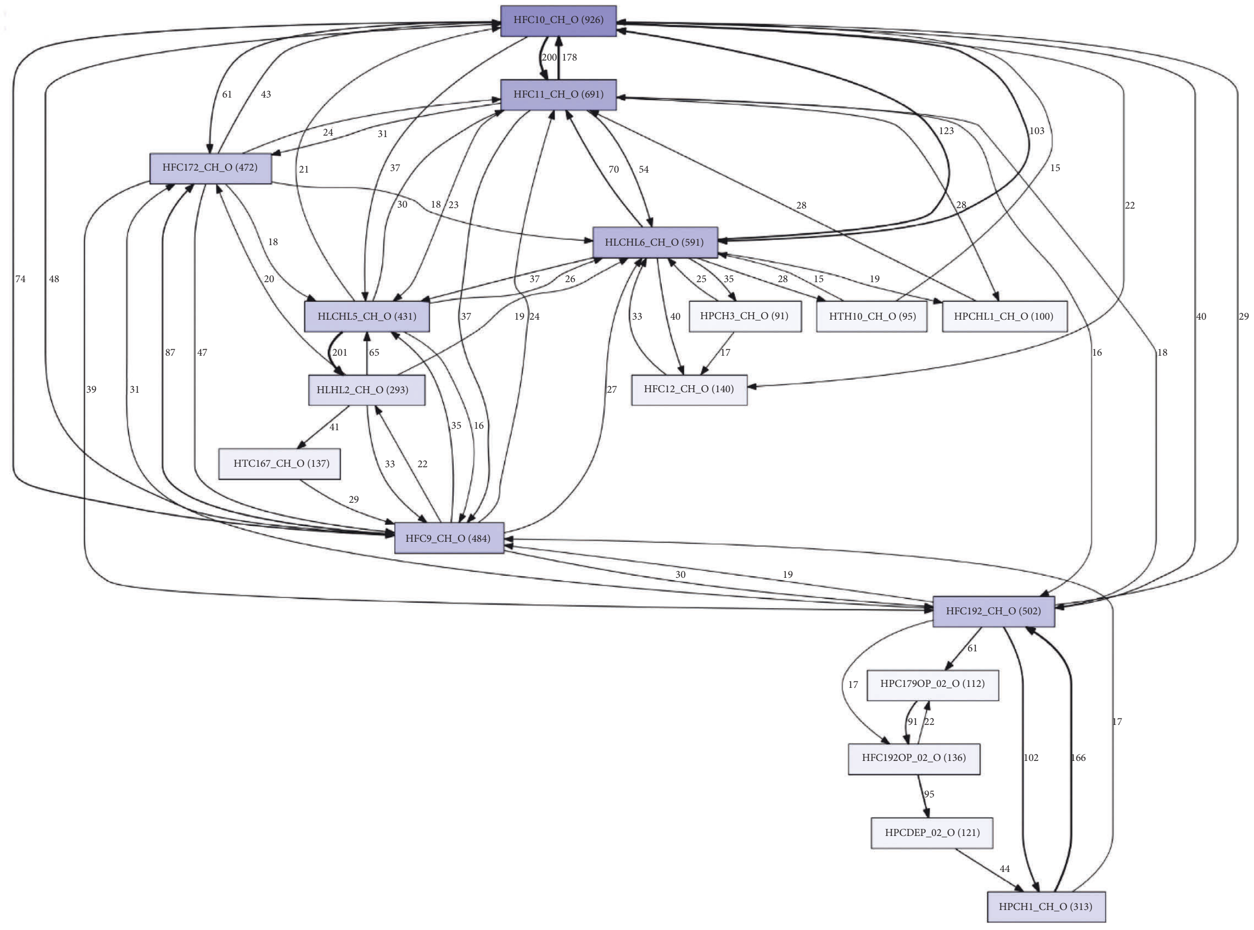


FIGURE 9: Series of operator actions. CH: isostripper, propane-depleting, and propane handling unit; 02: virtual unit.

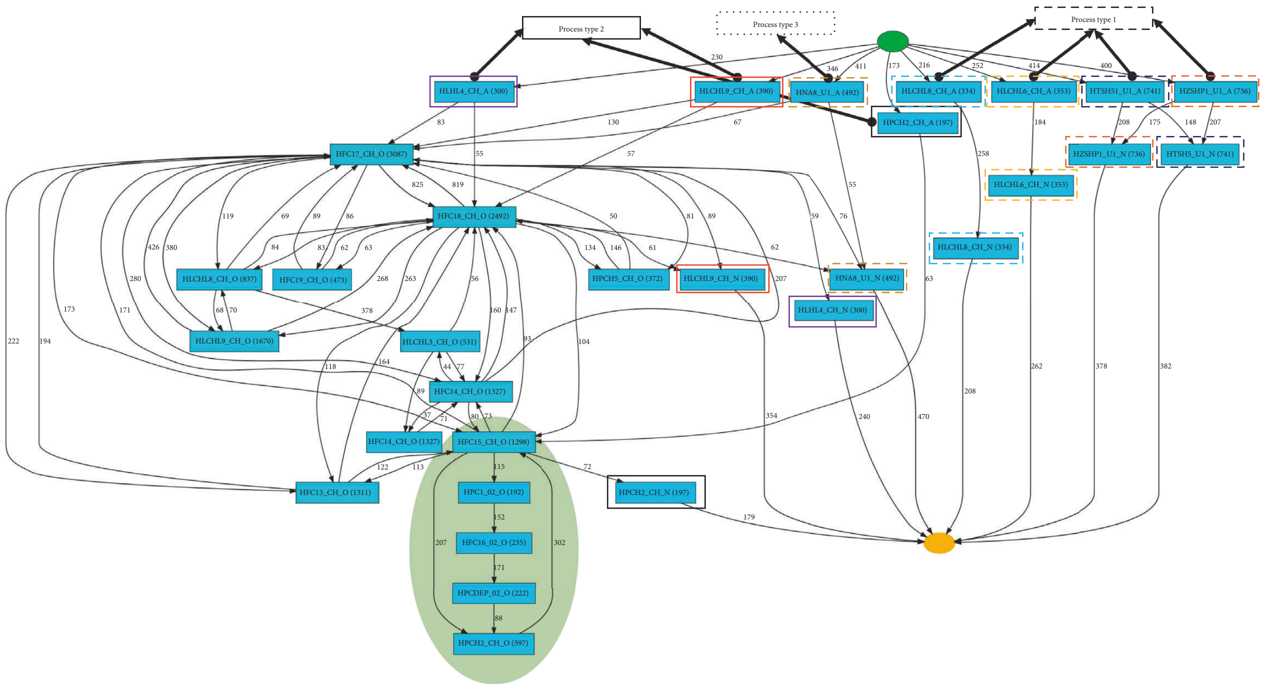


FIGURE 10: Alarm (A)-operator action (O)-return to normal (N) sequences. CH: isostripper, propane-depleting, and propane handling unit; U1: utility streams; 02: virtual unit.

same pump, they tend to be redundant and their definition should be revised by the process experts.

4. Conclusion

As industrial technologies are becoming more and more complex, the work of operators required to ensure the safe and optimal operation is getting increasingly challenging. With the introduction of the Industry 5.0 approach in progress, there is an emerging need for solutions to balance the productivity and efficiency with the life quality (work and home) of the workers, as well with the affection of the industry on society. One way to achieve this is to design alarm management systems based on tools that were developed to answer the challenges of the 4th Industrial Revolution (Industry 4.0). A properly built alarm management system can lower the workload of the operators significantly and reduce the probability of hazardous situations, affecting the environment (including civilians). The primary goal of this paper was to explore useful information from the historical data of industrial alarm management systems that can support the reduction of the operator workload and learn optimal operating strategies.

The paper proposed a process mining-based method to discover the fundamental relations between alarms and the related operator actions. Standard process mining techniques are not suitable for the analysis of historical process data of similar type. The paper demonstrated the benefits of the goal-oriented design of the log files that allows the extraction of information available to more effective alarm management and operator training.

The method was applied in the alarm management rationalization project of an industrial hydrofluoric acid alkylation plant. The project demonstrated that with the help of process mining, alarm signals could be rationalized; therefore, the work of the operators will become safer, as well as more effective, and last but not least, the workload has been decreased.

The discovered process models are easy to understand and provide some kind of improved digital visualization of alarms and operator actions. Lee et.al. summarized the applicability of digital twins in Industry 4.0-driven process safety management [26]. Our process mining-based method is also suitable to support some of the improvement actions collected in that study as a complementary tool. These improvement actions, related to alarms are: generate alarm signatures that can be useful in abnormal situation management, identify critical operator interventions, improve procedural risk assessments, and reduce the time and risk of errors during traditional risk assessment processes. They can support operator action-related tasks as well, namely, processing of procedures and operator actions: enhancing work design and operator performance, and representation and assessment of people and procedure related performance deviations and failures.

The gained information can be used to improve control systems, get a better insight into plant failure and behavior (process hazard analysis), review process safety incidents (incident investigation), and conduct what-if scenarios to understand how the plant may continue to run during unplanned maintenance or examining specific abnormal operation scenarios in more depth. It also gives the ability to assess initiating causes systematically and in an automated way based on historical data, due to the many failure modes of equipment that exist in a process plant. This is considered a key attribute to reduce resource intensive analysis.

Traditional hazard analysis processes have some shortcomings. A data science-based approach can address some of them, for example when there is a lack of

(i) depth of analysis
(ii) follow through to final consequences
(iii) completeness in identifying initiating causes and scenarios

The outcome of this study proved that the process mining-based analysis of events, along with the goal-oriented design of log files, should be added to the digital toolkit of process safety management.

Data Availability

The log data used to support the findings of this study have not been made available because of confidentiality reasons.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work has been implemented by the TKP2021-NVA-10 project with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development, and Innovation Fund, financed under the 2021 Thematic Excellence Programme funding scheme.

References

[1] E. Equipment, M. U. Association, E. Equipment, and M. U. A. Staff, *Alarm Systems: A Guide to Design, Management and Procurement*, EEMUA publication, London, England, 2015.
[2] G. Dorgo and J. Abonyi, "Sequence mining based alarm suppression," *IEEE Access*, vol. 6, Article ID 15379, 2018.
[3] G. Dorgo, P. Pigler, M. Haragovics, and J. Abonyi, "Learning operation strategies from alarm management systems by temporal pattern mining and deep learning," in *Proceedings of the 28th European Symposium on Computer Aided Process Engineering (ESCAPE28)*, A. Friedl, J. Klemes, S. Radl, P. S. Varbanov, and T. Wallek, Eds., pp. 1003–1008, Graz, Austria, June 2018.
[4] W. Hu, A. W. Al-Dabbagh, T. Chen, and S. L. Shah, "Process Discovery of Operator Actions in Response to Univariate Alarms," *IFAC-*, vol. 49, no. 7, pp. 1026–1031, 2016.
[5] D. S. Kim, H. Shinbo, and H. Yokota, "An alarm correlation algorithm for network management based on root cause analysis," in *Proceedings of the 13th International Conference on Advanced Communication Technology (ICACT2011)*, pp. 1233–1238, Gangwon-Do, South Korea, February 2011.

[6] N. A. Adnan, I. Izadi, and T. Chen, “On expected detection delays for alarm systems with deadbands and delay-timers,” *Journal of Process Control*, vol. 21, no. 9, pp. 1318–1331, 2011.

[7] H. Zang, F. Yang, and D. Huang, “Design and analysis of improved alarm delay-timers,” *IFAC-PapersOnLine*, vol. 48, no. 8, pp. 669–674, 2015.

[8] I. Izadi, S. L. Shah, D. S. Shook, S. R. Kondaveeti, and T. Chen, “A framework for optimal design of alarm systems,” *IFAC Proceedings Volumes*, vol. 42, no. 8, pp. 651–656, 2009.

[9] W. Hu, J. Wang, and T. Chen, “A new method to detect and quantify correlated alarms with occurrence delays,” *Computers & Chemical Engineering*, vol. 80, pp. 189–198, 2015.

[10] S. Lai and T. Chen, “A method for pattern mining in multiple alarm flood sequences,” *Chemical Engineering Research and Design*, vol. 117, pp. 831–839, 2017.

[11] G. Dorgo, P. Pigler, and J. Abonyi, “Understanding the importance of process alarms based on the analysis of deep recurrent neural networks trained for fault isolation,” *Journal of Chemometrics*, vol. 32, no. 4, Article ID e3006, 2018.

[12] G. Dorgo, A. Palazoglu, and J. Abonyi, “Decision trees for informative process alarm definition and alarm-based fault classification,” *Process Safety and Environmental Protection*, vol. 149, pp. 312–324, 2021.

[13] W. van der Aalst, *Data Science in Action*, pp. 3–23, Springer, Berlin, Heidelberg, 2016.

[14] J. Theis, W. L. Galanter, A. D. Boyd, and H. Darabi, “Improving the in-hospital mortality prediction of diabetes ICU patients using a process mining/deep learning architecture,” *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 1, pp. 388–399, 2022.

[15] P. Zerbino, A. Stefanini, and D. Aloini, “Process science in action: a literature review on process mining in business management,” *Technological Forecasting and Social Change*, vol. 172, Article ID 121021, 2021.

[16] M. Werner, M. Wiese, and A. Maas, “Embedding process mining into financial statement audits,” *International Journal of Accounting Information Systems*, vol. 41, Article ID 100514, 2021.

[17] R. Cerezo, A. Bogarin, M. Esteban, and C. Romero, “Process mining for self-regulated learning assessment in e-learning,” *Journal of Computing in Higher Education*, vol. 32, no. 1, pp. 74–88, 2020.

[18] V. Leno, A. Polyvyanyy, M. Dumas, M. La Rosa, and F. M. Maggi, “Robotic process mining: vision and challenges,” *Business & Information Systems Engineering*, vol. 63, no. 3, pp. 301–314, 2021.

[19] C. D. S. Garcia, A. Meincheim, E. R. Faria Junior et al., “Process mining techniques and applications – a systematic mapping study,” *Expert Systems with Applications*, vol. 133, pp. 260–295, 2019.

[20] R. E. Kondo, E. de F R Loures, and E. A. P. Santos, “Process mining for alarm rationalization and fault patterns identification,” in *Proceedings of the 2012 IEEE 17th International Conference on Emerging Technologies Factory Automation (ETFA 2012)*, pp. 1–4, 2012.

[21] R. E. Kondo, E. de Freitas Rocha Loures, E. A. Portela Santos, and C. M. P. Braga, “Alarm rationalization based on process mining techniques,” *Advanced Materials Research*, vol. 1061-1062, pp. 1258–1265, 2014.

[22] W. M. P. W. D. Aalst, “A practitioner’s guide to process mining: limitations of the directly-follows graph,” *Procedia Computer Science*, vol. 164, pp. 321–328, 2019.

[23] J. R. Taylor, “Automated HAZOP revisited,” *Process Safety and Environmental Protection*, vol. 111, pp. 635–651, 2017.

[24] H. M. W. Verbeek, J. C. A. M. Buijs, B. v. Dongen, and W. M. P. V. D. Aalst, *XES, XESame, and ProM*, vol. 6, 2011.

[25] B. F. V. Dongen and W. M. P. V. D. Aalst, “EMiT: a process mining tool,” in *Applications and Theory of Petri Nets 2004*, J. Cortadella and W. Reisig, Eds., pp. 454–463, Springer, Berlin, Heidelberg, 2004.

[26] J. Lee, I. Cameron, and M. Hassall, “Improving process safety: what roles for Digitalization and Industry 4.0?” *Process Safety and Environmental Protection*, vol. 132, pp. 325–339, 2019.