A TRACE CLUSTERING FRAMEWORK FOR IMPROVING THE BEHAVIORAL AND STRUCTURAL QUALITY OF PROCESS MODELS IN PROCESS MINING

Mohammad Imran[1, 2], Maizatul Akmar Ismail[1*], Suraya Hamid[1]

[1]Department of Information Systems, Faculty of Computer Science and Information Technology, University of Malaya, 50603, Kuala Lumpur, Malaysia

[2]Department of Information Technology, Faculty of Information & Communication Technology, Balochistan University of Information Technology, Engineering & Management Sciences, 87300, Quetta, Pakistan

Email: Maizatul Akmar Ismail (maizatul@um.edu.my) Corresponding author

ABSTRACT

Process mining (PM) techniques are increasingly used to enhance operational procedures. However, applying PM to unstructured processes can result in complex process models that are difficult to interpret. Trace clustering is the most prevalent method for handling this complexity, but it has limitations in dealing with event logs that contain many activities with varied behaviours. In such cases, trace clustering can produce inaccurate process models that are expensive in terms of time performance. Therefore, it is crucial to develop a trace clustering solution that is optimal in terms of behavioural and structural quality of process models while being efficient in terms of time performance. In this study, we introduce a refined trace clustering framework with an integration of log abstraction and decomposition technique that improves the precision of process models by 38%, leading to a 40% increase in the f-score. The proposed framework also produces process models that are 38% simpler than those produced by baseline approaches. More importantly, our framework achieves a remarkable 89% improvement in time performance, making it a valuable contribution to the field of process mining. Future works include exploring the scalability of the proposed framework against a wider range of complex event logs and testing the framework to validate its effectiveness in practical applications.

Keywords: Complexity, complex process models, event log granularity, process mining, trace clustering framework

1.0. INTRODUCTION

Process mining (PM) combines data mining and business process management (BPM) techniques to analyse and optimize business processes. The basis of such analysis is the event logs generated by the information systems. With event logs in hand, process mining starts with process discovery [1], [2], a sub-type of process mining used to discover the visual representation of processes called process models [3],[4]. The discovered process models are then used for the identification of process inefficiencies, their root causes, and their impact on key performance metrics. Process discovery is the basis of subsequent process mining techniques, i.e., process conformance checking and process enhancement [1]. Given that processes are initially modelled using process discovery techniques, it follows that incorrect process discovery would produce inaccurate results for the subsequent process analysis. Therefore, valid process discovery is essential to reach a valid conclusion about process performance and compliance.

Traditionally, process discovery works very well for structured processes. However, real-world processes are not straightforward and contain flexible and ad-hoc behaviours. Application of process discovery on such unstructured processes results in complex process models, which introduces the challenge of interpreting process models [5]–[7]. An example of such scenarios is the process of an organization where no restriction is imposed on the order of activities to follow. In case of such a flexible process, employees may use any sequence and combination of activities. When such a process is executed, it generates a variety of process execution traces with an arbitrary number and order of activities. Such a flexible process is referred to as complex or unstructured process in process mining. Applying process discovery on such event logs produces a spaghetti-like representation called complex process model [8]–[10].

According to [11] and [12], as cited in [13], the complexity and density of a process model are known to have an impact on a human analyst's capacity to comprehend it. Therefore, to understand and improve the process, the resulting process models should not be overly complex. The complexity issue has been addressed by several researchers using four primary approaches, clustering, abstraction, filtration, and pattern mining; however, trace clustering is the most popular approach [14]. Abstraction approach can simplify complex event data by reducing the size of event logs and focusing on important aspects of the process [15]. However, using abstraction approach ignores less frequent behaviours and the experience of the abstractor also influences the results [16]. A lot of candidate patterns are generated by such a method, which makes

it an expensive approach for managing complexity. Filtration approach deals with complexity using the notion of noise in the event log [17], [18]. By removing noise, a reduction in the size of event logs is observed, thereby making the analysis of the process more manageable. However, throughout process mining literature the notion of noise is typically attributed to less frequent activities in the event log and the use of filtration approach discards such infrequent activities [14]. Commonly, non-compliant behaviours occur infrequently; therefore, removal of infrequent behaviour also implies complicating the detection of incompliant behaviours and thereby threatening the validity of the subsequent process mining step such as conformance checking. On the other hand, pattern mining approaches can help in understanding the behaviour of the process by focusing on specific patterns [19], however, they depend on the modeler's decision to include or exclude specific behavioural patterns and are only practical when end-to-end model discovery is not a concern [14]. In the Trace clustering approach traces with similar behaviours are grouped together into clusters, providing a way to reduce the size of event logs and focus on important aspects of the process. Compared to other approaches, trace clustering techniques are superior in terms of retaining maximum possible behaviour in the event log, and all behaviours are included in process discovery regardless of it being frequent or infrequent [14]. In the trace clustering approach, process instances are divided into clusters based on the features of process instances. Such features are referred to as "trace profiles" [20]. For each set of profile-specific clusters, a sub-model is generated which is simpler than a single and all-inclusive process model. The quality of such process models is commonly evaluated using two dimensions, behavioural complexity, and structural complexity [14]. Behavioural complexity consists of fitness, precision, and f-score metrics, where fitness is used to measure the capability of the process model to replay the behaviours found in the event log while precision quantifies the specificity and accuracy of the process model. On the other hand, structural complexity is used to assess the complexity and the simplicity of the resulting process models. A process model with low precision contains highly generalized behaviour which does not exist in reality. Since there exists a trade-off between fitness and precision, having an optimal balancing between two is a desired characteristic [21], [22].

An extensive literature review of the complexity issue [14] and the experimental evaluation reveals that existing trace clustering approaches help reduce the model complexity when event logs contain trace level variation. However, when the event log contains a high number of activities with varying behaviours, trace clustering does not help reduce the complexity of process models; instead, some level of improvement in fitness quality is observed. Secondly, such techniques remain expensive in terms of time performance. Improving just the fitness or precision quality is not enough rather striking a balance between these two model quality metrics is essential [22]. This problem is prevalent throughout the process mining literature [14], [15], [23]. Therefore, there is a need and scope for trace clustering techniques that generate simpler process models, and along with the fitness, the precision of process models is also improved. Finally, considering the expensiveness of the existing approaches, it is also important that the resulting trace clustering technique performs efficiently.

Keeping in view the identified limitations, we present an enhanced trace clustering framework that not only produces more precise but also yields simpler process models. Furthermore, the proposed framework exhibits a notable time reduction of 31 minutes in comparison to the two baseline studies [20] and [24], which required 201 minutes and 306 minutes respectively. Consequently, the proposed framework not only enhances the overall performance but also significantly improves the time efficiency of the trace clustering solution. The rest of the paper is structured as follows:

In section 2, we present the basic terminologies, a background of complexity issue in process mining, the motivation behind trace clustering, and related works. In section 3, the proposed trace clustering framework is presented. In section 4, the experimental setup is elaborated. In section 5, the results are presented while in section 6, the significance and implications of results are discussed along with the limitations of the proposed framework. Finally, in section 7, we conclude our findings.

2.0. BACKGROUND AND RELATED WORK

In this section, we first present the background of the complexity issue in process mining. Then, the basic terminologies used throughout the article are elaborated. Later the trace clustering concept and the related trace clustering techniques used for complexity reduction are discussed in the related works sub-section.

The information systems of an organization execute business processes. These business processes leave their footprints as event logs [25]. Process mining techniques use such event logs to discover the actual execution of the business process. The extracted business process models are then used for several purposes, such as checking the compliance of business processes against standard operating procedures and finding process optimization opportunities [26]. The real-world processes are not straightforward, as they are characterized by flexibility. Execution of such processes results in varying

process behaviours recorded in the event logs. In process mining domain, such flexible processes are referred to as complex or unstructured processes [14]. Application of process mining over such processes results in so-called spaghetti process models, which are imprecise on the one hand and challenging to comprehend as well [27].

Researchers have put forward several strategies to deal with complexity problem. These strategies include changing the abstraction level of the event logs [28], behavioural filtration [29], restriction of process discovery to specific behavioural patterns [30] and clustering of process instances into sub-logs [31]. The clustering approach, however, has a significant advantage in retaining maximum information and considers all behaviours as observed in the event log [14].

2.1 Basic terminologies

2.1.1 Event

An event is a tuple that represents a single activity executed in a process. Each event tuple may contain an activity name, human or machine resource involved in activity execution, and the timestamp of its execution. In the event log of an example process in Table I, each row represents the execution of an event which contains an activity name, the human or machine resource who performed the activity, and the timestamp at which the activity was executed. The sample event log in Table I contains twelve events with five unique activities.

2.1.2 Trace

A trace or process instance is a footprint of single process execution. Each trace is a combination of multiple events. The traces are represented by a unique Case ID which acts as a unique identifier of the trace in the event log. The example event log shown in Table I contains three unique traces where trace "1" executed four events, trace "2" executed three events, and in the context of trace 3, five events were executed.

2.1.3 Event log

An event log is a collection of traces or process instances that denotes the multiple executions of the process. Each row of the event log represents an execution of an event. Table I depicts the sample event log containing three traces and twelve events executed by five employees between 1st December 2022 and 4th of December 2022.

Table I. Event log of a sample product order and dispatch process

Case ID	Activity	Resource	Timestamp
1	Receive order	Emp 1	1-12-2022 11:00
1	Check Inventory	Emp 2	2-12-2022 09:17
1	Check payment status	Emp 3	4-12-2022 13:15
1	Dispatch	Emp 4	4-12-2022 14:45
2	Receive order	Emp 1	3-12-2022 17:01
2	Check Inventory	Emp 2	4-12-2022 12:25
2	Cancel order	Emp 5	4-12-2022 13:45
3	Receive order	Emp 1	1-12-2022 14:53
3	Check Inventory	Emp 2	2-12-2022 15:32
3	Check Inventory	Emp 2	3-12-2022 09:02
3	Check payment status	Emp 3	4-12-2022 16:25
3	Dispatch	Emp 4	4-12-2022 16:58

2.2 Trace clustering

An event log contains varying process execution behaviours where some behaviours are homogenous such as executing similar activity sequences [32]. On the other hand, heterogenous behaviours also exist where some of the process execution behaviours remain dissimilar to others [33]. In sample trace <ABDCE>, activity "A" is directly followed by B and contains five activities. In another sample trace <ADBCE>, activity "A" is directly followed by activity "D" and contains a total of five activities. The directly-follows relation between two activities is an example of process execution behaviour. Both traces have the same number of events; therefore, they would be placed in the same cluster if the clustering criterion were based solely on this metric. However, if the context of activity execution behaviours is included, such as which activities follow activity A, both the traces fall into separate categories because of heterogenous behaviours. In this way, traces can be clustered into smaller groups based on a wide variety of such characteristics to reduce the complexity of the process models.

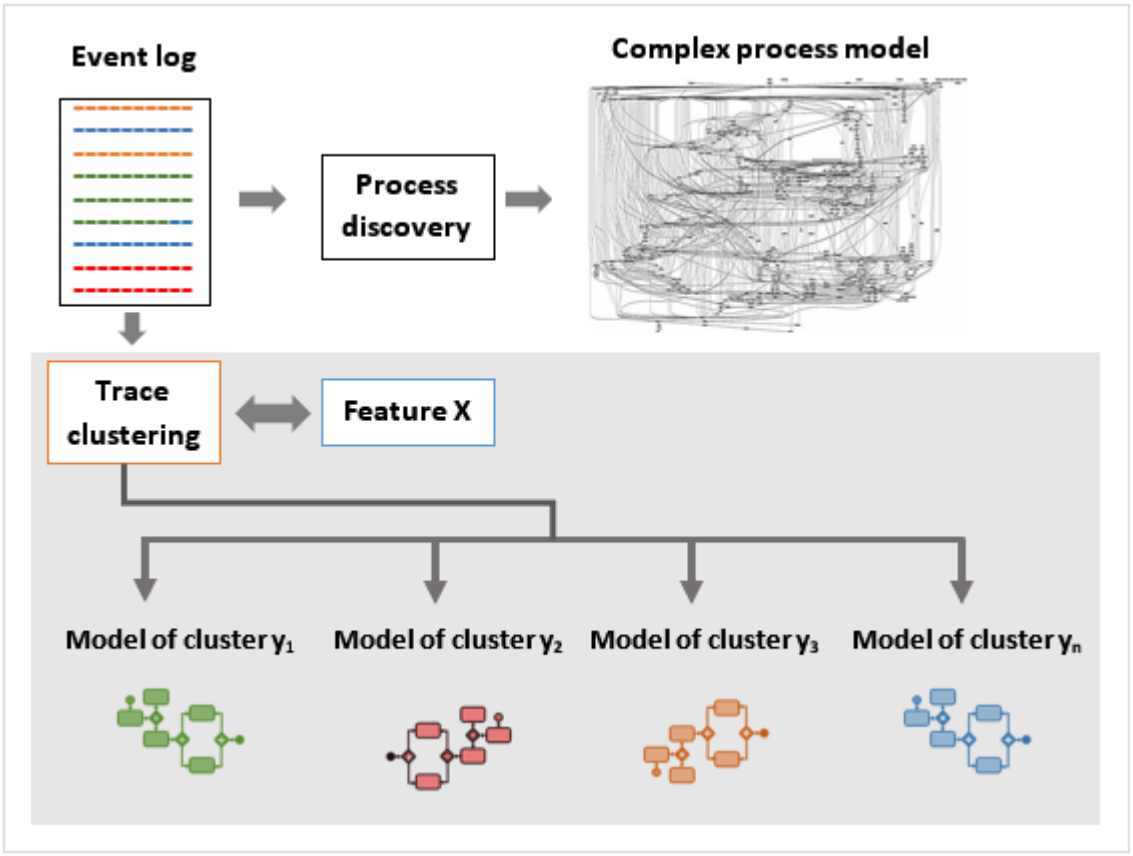

Fig. 1. Trace clustering concept

Real-life processes may contain several different process execution behaviours. Generating a single process model from an event log that contains high variation results in a spaghetti-like representation, which adds to the model's complexity. [34]. Therefore, based on the characteristics of the traces, the event log is divided into groups of similar behaviours. Such a process of grouping traces based on similarity or dissimilarity is known as Trace clustering [35]. Fig. 1 represents the abstract concept of trace clustering based on feature profiles.

Another motivation behind performing trace clustering is the fine-granular level of details about process execution in the event logs. Consider example trace <ABDDDCCGTFHYTJDDDIIIIIIMMMMYYYYYYYY> containing thirty-five activities. Discovering process models from event logs of such fine-grained nature results in a model containing a high number of nodes and edges, which translates to the complexity of process models. Moreover, the variation among the traces introduces yet another level of complexity.

2.3 Related works

According to a comprehensive analysis of the available methods for simplifying complex processes, trace clustering is still the best option for process mining [14]. Some researchers concentrated on activity clustering, while others employed guided trace clustering and the trace-profiles-based clustering method. In the following, we present an overview of all three methods.

In order to deal with fine-grained event logs, a trace segmentation approach was utilized [36]. The trace segmentation approach utilizes the concept of co-occurrence of activities, which refers to the high frequency of two activities appearing together in the event log. As an example, if activity A and C frequently occur together more frequently than activity G and Y, then activity A and C are considered segments of some higher level of activity and clustered together and later abstracted as a single activity to simplify complex process. In order to deal with the trace level variation, [35], [37], and [38] utilized the activity sequences to cluster the traces into homogenous subsets. A sequence of activities refers to an event log pattern where two or more than two activities occur together in a in a direct succession to each other. Such as in example event log [<A, B, C, D>2, <A, B, C, F, G, H, I, Z>20, <A, D, F, G, H, I,>7, <A, C, D, X>2], where the subscript number indicates the number of times a trace is found in the event log. It can be observed that the sequence "F, G, H, I" frequently appears in the event log with a frequency of 27 in the log (20 times in second trace variant and 7 times in third trace variant). Moreover, the sequence, "A, B, C" appears 22 times (2 times in first trace variant and 20 times in second trace variant). Therefore, more frequent sequences are abstracted into a single activity X as in <X, C, D>, <X1, X2, Z> to reduce the complexity. Instead of co-occurrence or activity sequences, activity hierarchies were utilized for clustering the traces by [39]. Activity hierarchies are a way to organize activities in a process into a hierarchical structure, with high-level activities at the top and low-level activities at the bottom. These hierarchies can be useful for abstraction in process mining to guide the abstraction process [40]. In this form of abstraction, only high-level activities are focused on which are the result of the aggregation of multiple lower-level activities. For example, activities "receive order", "check inventory", and "process payment" are part of higher-level task activity called "order processing". So, these low-level tasks are aggregated into single high-level activity to simplify the complex process [41]. Similarly, researchers in [42] also used activity hierarchies to conduct research, but their experiments focused on structured processes rather than unstructured ones. Attribute-based clustering was proposed in [43]; however, the authors did not perform an experimental evaluation of their method. Using the notion of time interval, researchers in [32] proposed grouping activities that occurred within brief time intervals to construct simpler models.

In structured processes there are clearly defined sets of tasks or steps that need to be completed in a specific order. Examples of structured processes include manufacturing assembly lines, loan approval processes, and order processing systems. On the other hand, unstructured processes do not have a clearly defined set of tasks or steps, and they may involve a high degree of variability and uncertainty. These processes are often more flexible and adaptable than structured processes, but they can also be more difficult to manage and analyse. Examples of unstructured processes include customer service interactions and healthcare processes. The application of co-occurrence of activities [36], activity sequences [35], [37], [38], and activity hierarchies [39], [42] are well suited for structured processes where explicit hierarchies and explicit high-level counterpart is available. The research works have been mainly applied their methodology to processes that had a clear and well-defined set of tasks or steps to be completed. This contrasts with unstructured processes, which may not have a clearly defined set of tasks or steps, making them more challenging to analyse using traditional process mining techniques.

In the context of guided clustering approaches, [44] and [45] proposed to cluster traces based on specific trace patterns that deviate from normal behaviour. Likewise, the researchers in [34] employed a set of constraints over clustering process. They [34] first extracted the process model having the highest prevalence and then clustered the traces guided by this new process model. With an aim to get more fitting process models, the fitness quality of the process model was used to guide the trace clustering process [46]. However, focusing only on increasing the fitness quality reduces the precision quality of the process models, which consequently increases false positives in process models, i.e., behaviours that do not exist in reality. Therefore, it can be argued that the precision dimension is equally significant and should not be ignored. Throughout the process mining literature, researchers [22], [47], [48] have repeatedly highlighted the significance of a balance between these two model quality measures. Domain knowledge, such as disease occurrence patterns, was employed by [49] to guide the trace clustering process. Traces were grouped based on high precision between process models and the selected patterns. However, the need for existing patterns limits the generalizability of their method.

In the context of clustering the traces based on trace features, researchers in [50], [51], [20], [24], and [52] proposed to use several distinct event log features to group similar traces. The main idea behind trace-features-based clustering was to project the event log into a vector space representation called trace profiles. Each trace profile was a collection of features that represent the certain different perspective of the traces. Based on feature profiles, the traces are clustered into similar sub-groups. One such example of a feature is the frequency of a sequential relation A > B in the event log. In context of feature-based trace clustering, some researchers [50], [51], [20], [24], [52] we successful in enhancing the fitness quality of the process models. However, equally vital process perspective, the model complexity was disregarded. Secondly, clustering traces based on trace-level features only improves the fitness quality of process models, negatively impacting the model precision. Moreover, trace clustering solution that relies on several feature profiles with a range of additional sub-features is costly in terms of time performance and does not ensure any reduction in process model complexity. Another problem with feature-profile-based methods [20], [50], [51] is that they often ignore feature frequencies and employ a frequency-isolated viewpoint. According to [14], the number of activities and their cyclical nature are also one of the major contributors to process model complexity. Therefore, frequencies should also be taken into account.

The review of the relevant literature indicates that several of the currently available trace clustering methods effectively divide the log into subsets. Even though progress was made in lowering the complexity brought about by trace-level variance, the complexity introduced by the fine granularity of the event log received comparatively less attention. On the other hand, while evaluating the process behavioural quality of the process models, the focus has remained on fitness quality, while the precision dimension has been disregarded. It can be argued that although the process model should conform to the reference models; however, for an optimal trace clustering solution, it is essential to strike a balance between fitness, precision, and complexity. Without striking a balance between these quality parameters, actionable knowledge discovery will not be possible [53], [54].

3.0. PROPOSED TRACE CLUSTERING FRAMEWORK

In this section, we present the proposed trace clustering framework, which is illustrated in Fig. 2. The dotted colour lines in the framework represent the elements of the existing trace clustering framework starting with feature profile extraction.

An N number of feature profiles can be extracted based on the availability of additional data attributes in the event log. Then based on each feature profile, the event log is clustered using a clustering algorithm which results in M number of trace clusters for N number of features profiles. For each cluster M, a process model P is generated, representing a unique perspective on the event log. To assess the quality of trace clustering, having several process models in hand, mean fitness, mean precision, mean f-score and mean simplicity (complexity assessment) values are calculated.

In the refined trace clustering framework, we have proposed an event log abstraction strategy that first assesses the extent to which certain elements introduce complexity in the process model. Then based on the threshold, abstraction is applied to such elements by mapping them with the corresponding high-level counterparts in the event log. In this way, the granularity of the existing event log is improved from fine-granular quality to course-granular quality. Such an abstraction helps in reducing the complexity of the process models without sacrificing the context of the activity execution.

However, despite a reduction in complexity, abstraction did not improve the precision quality of the process models. Therefore, we proposed an event log decomposition strategy dividing the event log into logical subsets. First, such a feature is identified that can logically divide the process into logical sub-processes. Then based on that feature, the event log is decomposed into sub-logs.

After performing abstraction and decomposition, feature profiles are extracted from the event log, and the rest of the framework steps are applied. Decomposition of the event logs into logical subsets results in precise process models. Moreover, it also enhanced the simplicity of the process models.

3.1 Abstraction

Since the fine-granular level of activities in the process models introduces the complexity, the higher the number of activities in the process model, the higher the complexity. Therefore, to reduce the complexity of the event log, we propose mapping fine-grained activities of the event log against their high-level counterpart within the event log. To retain the context of activity execution, the most relevant feature was used as a high-level counterpart. However, mapping

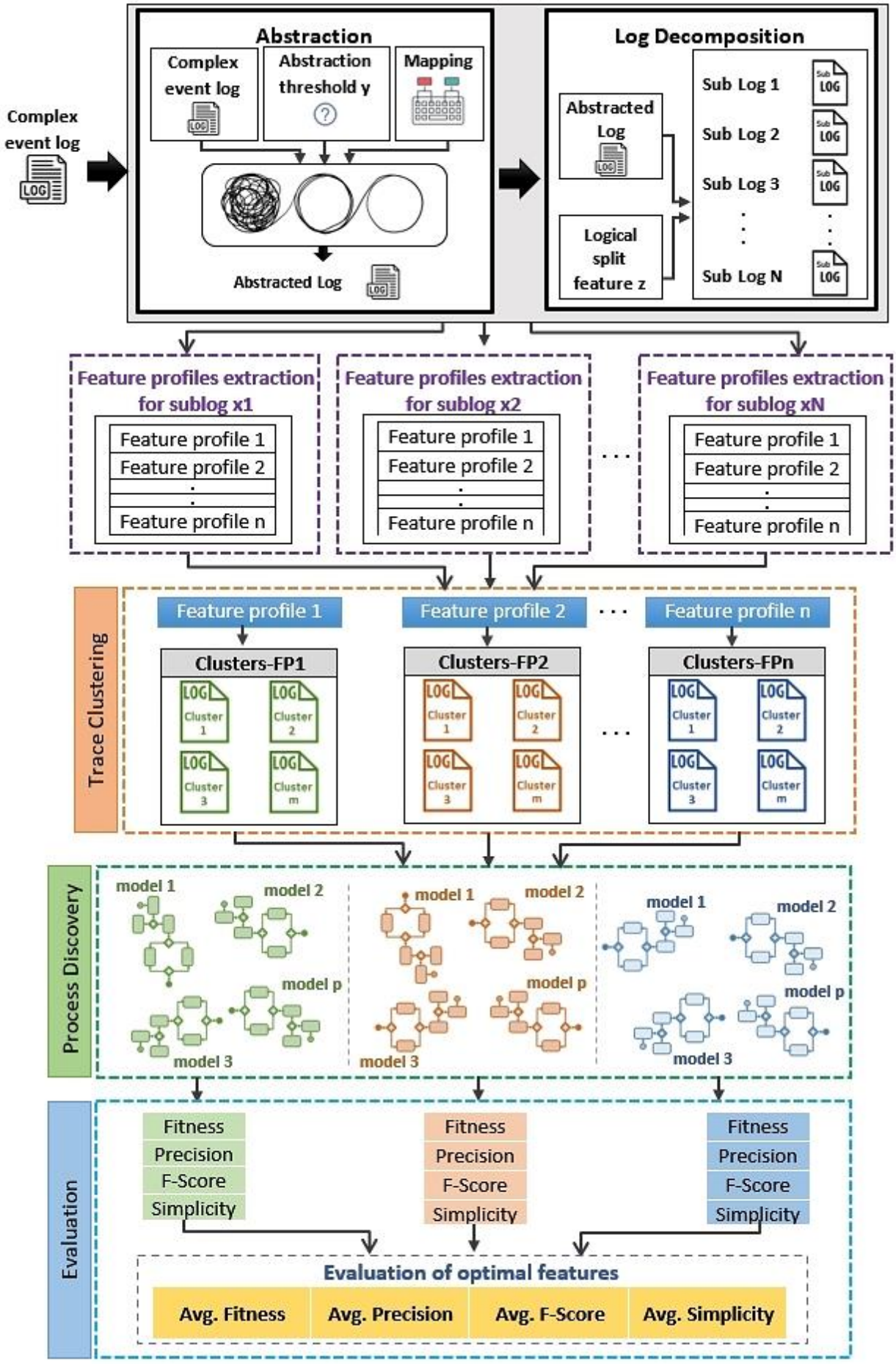

Fig. 2. Proposed framework for trace clustering with abstraction and decomposition.

all activities to their high-level counterpart will fully remove the activity execution details. Therefore, we employed a threshold on the abstraction level. Even though abstraction improves the interpretability of the process models, it is also important to consider the trade-off between precision and abstraction. The higher levels of log abstraction can produce simpler process models but may sacrifice precision. On the other hand, if event log contains excessive details about process execution, the resulting process models become more complex due to the overwhelming number of individual actions that need to be incorporated into the process model. Therefore, based on the granularity of the event logs in hand, an abstraction threshold must be set in such a way that a balance between the precision and model simplicity is achieved. We experimented with several thresholds values and based on the evaluation, the threshold that provided the best balance among precision and simplicity was set as 20. This threshold acts as the cut-off value on the mapping between fine-granular and coarse-granular activity.

3.2 Logical decomposition

Despite the reduction in complexity, the abstraction does not help in producing in improving the precision of the process models. Therefore, we employ the decomposition of the event log into logical subsets before performing trace clustering. Like the abstraction strategy, to retain the context of the activity execution, the decomposition of the event log is also guided by the features present inside the event log. Event log features can be leveraged to partition event logs into meaningful subsets, which can divide the logs in a way such that each split contains a sub-process comprising of related activities. Within the framework of this study, we employed an event log consisting of disease treatment procedures from a hospital, encompassing various activities conducted during the treatment processes for different diseases. The diagnostic and treatment procedures for each disease involve varying activities. For instance, the activities for treating stomach cancer and ovarian cancer differ due to the distinct types of cancer affecting different organs. Additionally, these activities may significantly differ on various other factors such as the stage of the disease, the overall health of the patient, and individual considerations. As the hospital log's "disease identifier" feature contained contextual information regarding activity execution, this specific feature was employed as the logical splitting criterion, denoted as feature Z in log decomposition part of Fig. 2. The number of decomposed partitions of the event logs depends upon the number of unique elements of the feature used for decomposition.

3.3 Extraction of features for clustering

After the abstraction and decomposition process, for each decomposed event log, several features are extracted from the event log, i.e., the trace profiles. For experimental purposes, we used ten feature profiles presented in Table II. Each feature profile is an x, y vector of several sub-features representing a vector space representation of trace against that specific feature profile.

Table II. Feature profiles used for clustering.

Occurrence	Feature Specificity	Feature profiles	Description
Frequency-based	Universal	Activity	Trace-level frequency of activities
		Activity duration	Trace-level duration of activities
		Activity transition	Trace-Level frequency of transition between activity x and y
		Case duration	Overall duration of each trace
		Department	Trace level frequency of department
		Department transition	Trace-Level frequency of transition between department x and y
		Resource	Frequency of human resource involved in activity execution
Real values		Variant	Unique trace variant
	Log specific	Age	Age of patients
		Patient Type	Type of patient (emergency or normal)

An excerpt of an example feature profile is presented in Table III which is a transition profile. A transition refers to a kind of process execution behaviour observed in the event log which represents a certain sequence of activities performed towards execution of the process. In a sample trace <A, B, C> of process execution instance it can be observed that in total three activities were performed towards execution of the process where process started with activity A and ended with activity C. Moreover, two process execution behaviours can be observed in this trace A > B and B > C. The transition A > B implies a behaviour A is followed by B was found in the trace and transition B > C shows that another behaviour B is followed by C was also found in the event log. However, since an event log contains several executions of process instances i.e., trace, therefore there may be several other traces that contain such transitions. Moreover, other traces may also contain several other such features which homogenizes or distinguishes it from other traces. This triggers the need of a feature profile matrix where each row in feature profile matrix represents the frequencies of a particular feature (in column) against each trace. The traces are uniquely identified by a Case ID. Consider transition feature B > C for trace number 7 in Table III, which indicates that for trace number 7, the transition feature B to C exists with a frequency of 20, which implies that in the event log, activity B was followed by activity C for a total of twenty times.

Activity profile is yet another feature profile which is used to distinguish the traces based on the activities that contain. While other approaches do ignore the frequencies, we consider the frequencies as an essential parameter for clustering, model complexity, and model evaluation. Merely considering the binary occurrence of the trace features does not represent the true impact of that feature on the trace. For example, in cased of activity profile, two sample traces, <ABBBBCCCDDEEE> and <ABCDE>, are considered equal in existing trace clustering approaches because both traces executed exactly the same kind of activities i.e., A, B, C, D and E. However, it can be observed that in the context of frequencies, both the traces differ from each other. Therefore, ignoring frequencies discards the context in which the activity was executed.

The usage of frequencies has significance in several other contexts too. Considering the real-life dataset of a patient treatment process in a hospital (also used in this study) where patient A's treatment process contains five activities. On the other hand, despite the same activities followed for patient B, some of the activities are repeated more than once, such as reinvestigating the diagnostics tests. Ignoring such frequencies will result in clustering two patients together, which is incorrect and results in a loss of activity execution context where patient B's treatment process was more rigorous than patient A. It is evident that feature frequencies do have an impact on clustering results. Therefore, considering this rationale, we do consider feature frequencies in trace profiles instead of binary decisions on the presence of a certain feature in a trace.

Table III. An excerpt of a feature profile

Case ID	***A > B***	***B > C***	***A > D***	**….**	***ABC > XYZ***
0	*1*	*3*	*2*	-	*2*
1	*0*	*16*	*8*	-	*1*
2	*0*	*0*	*0*	-	*0*
3	*0*	*0*	*0*	-	*0*
4	*0*	*8*	*1*	-	*3*
5	*0*	*1*	*0*	-	*0*
6	*0*	*2*	*0*	-	*1*
7	*1*	*20*	*6*	-	*2*
-	-	-	-	-	-
12567	*0*	*5*	*0*	-	*2*
12568	*2*	*50*	*3*	-	*26*

Since the features vary from log to log, few of these features are specific to the dataset in hand, while others are prevalent to every process mining dataset. Table II shows all the features we used for trace clustering. These features were extracted from event log with the help of Microsoft Excel.

3.4 Trace clustering

In this stage, the traces are clustered by the clustering algorithm based on feature profiles. We used ten feature profiles that were extracted in the previous step to form trace clusters. These feature profiles are listed in Table II. The 'Activity profile' is combination of activity feature matrix which represents the frequencies of certain activities appearing in the traces. Using activity profile, the traces can be clustered based on the frequencies of activities. Similarly, the feature matrix 'Activity duration profile' and 'case duration profile' are combination of the features that represent the time taken

to complete each activity within a trace and overall time taken by each case respectively. These features profiles help form, clusters based on temporal properties of the traces. The 'activity transition profile' is combination of features that contain the frequencies of directly follows relation between the pair of activities in the event log. Such a relation helps in clustering event log based on the activity sequences that contain. The 'department profile' represents the frequency of involvement of certain department with a trace which can be used to cluster the traces. Similarly, the 'department transition' profile contains the features representation of secession frequency between any two departments. The resource in process mining traditionally refers to human resource or machine resource involved in execution of certain activity of the process. The resource represents the activity handover and social network of resources towards execution of certain process. Therefore 'resource profile' is combination of feature frequencies of each resource involved in the execution of certain frequencies.

Similar to the 'transition profile' which represents the features related to sequential appearance of the two activities in the traces, the 'variant profile' is used is used to distinguish the traces based on the sequences of activities they executed. Compared with transitions profile, however, there is no limit on number of activities appearing in the sequence. Based on the sequence, each variant is assigned a unique identifier and the frequency of such identifier is used as feature profile. Apart from universal event log features we also included some log specific features profiles such a 'Age profile' and 'Patient type profile'. The Age profile is used to cluster the traces based on the age of the patients involved. On the other hand, the 'Patient type profile' is used to categorize the patients based on severity of disease such as emergency patient or normal patient as the attention procedures and sequence of activities executed to treat patients with emergency differ than those who require normal priority. Principally, any clustering algorithm can be used in this stage to cluster the event log. However, since K-Means can scale well to large datasets and feature frequencies have been used in the feature profiles, we used K-Means to cluster the traces. The maximum number of clusters can be decided by using silhouette coefficients and elbow method.

3.5 Process Discovery

Once the clustering process is complete, a process model is discovered for each cluster generated in trace clustering step. The selection of the process discovery algorithm is an important parameter here. Principally, any process discovery algorithm can be used such as Alpha miner, Heuristic miner, Fuzzy miner, or the Inductive miner. However, the limitations of the process discovery algorithms should be kept in mind when performing process discovery. The alpha miner cannot discover short loops; therefore, the models will not be accurate. Heuristic miner used a behaviour filtration threshold; therefore, the resulting model will always leave out infrequent behaviour which leads to inaccurate and complex process models. An important property of process models is soundness which ensures that output model provides some guarantees such as it must be complete and able be capture all possible ways in which the process can be executed. This helps to ensure that the model is accurate and reliable. Both the alpha miner and heuristic miner produce unsound process models which results in inaccurate representation of reality. Fuzzy miner also uses a similar behaviour filtration approach on based on activity and relation frequencies however it lacks formal process semantics, i.e., no difference can be made between a choice and a join or split, thereby removing the capability to evaluate the behavioural quality of the process models. Another important point to consider is that in the context of computational expensiveness, process discovery is considered a computationally expensive task in process mining which is further escalated due to fine granularity of the event log. A high number of places and transitions in source event log creates a performance overhead due to an increase in calculations of relations among activities. However, due to log decomposition, abstraction and clustering, the number of activities per model thereby downscaling the number of paths required to model. In this way, abstraction and decomposition do implicitly resolve the issue of computational expensiveness of the mining algorithm. Compared to all three other process discovery algorithms, the Inductive miner does not leave out infrequent behaviour and always produces sound process models. Secondly, since the inductive miner produces petrinet modelling notation, which enables the evaluation of the discovered process models' quality.

Compared with alternate process discovery algorithms, having the properties of accuracy, soundness and enabling of model quality assessment makes inductive miner a suitable candidate for process discovery. Therefore, we selected the Inductive miner as the model discovery algorithm.

3.6 Evaluating the behavioural and structural quality

In this stage, the behavioural and structural quality of the resulting process models are measured. We used well-known fitness, precision, and f-score metrics for the evaluation of the behavioural quality of the process models [14]. Moreover, for assessment of the structural complexity and simplicity of the resulting process models, inverse arc degree [14] was

used. The inverse arc degree is the average number of input and output arcs connected to each transition/ place of a process model. The more connections in the process model, the more complex a model becomes. To provide a comprehensive understanding of model complexity reduction, we incorporated two supplementary metrics for evaluating structural quality: model density and places count. Model density refers to the proportion of total arcs in the model compared to the maximum possible number of arcs [17], [38]. A higher density signifies a more intricate process model. Moreover, places count indicates how many individual components make up the final process model [34]. Increased number of nodes represent an increase complexity of process models. A greater number of nodes indicates increased complexity.

3.7 Selection of optimal clustering solution

Based on the evaluation results the optimal settings for trace clustering are selected. Optimal settings refer to the situation where trace clustering results in highest possible fitness, precision, and simplicity of process models, and the overall time taken for trace clustering and model evaluation is reduced. Such an optimal setting is deemed as recommended trace clustering solution.

4.0. EXPERIMENTAL SETUP

We used the following setup for the evaluation and experimentation of the proposed framework:

In the experiments, we compared the results of our proposed framework against the approaches of Jablonski et al. and Faizan et al. who performs trace clustering with any abstraction and decomposition.

As input, we used publicly available real-life event logs of the patient treatment process of a Dutch Academic Hospital [55]. The selection of this dataset was motivated by its frequent usage in process mining to evaluate process complexity management techniques. The average number of events per trace (131) in our source event log already indicated a high level of fine granularity of the event log, which translates to the complexity of the process models. Since the event log is highly granular, in the first phase, we applied abstraction over the event log where low-level high-level activities are remodelled with high-level counterpart. However merely abstracting all the activities produces highly imprecise process models. Therefore, a threshold value is employed which is guided by precision and model complexity. This threshold value is tested against multiple abstraction thresholds in order to achieve a balance between the precision and model complexity.

To further improve model precision, in second phase, log decomposition is applied over abstracted event logs which is guided by event log features. We used “disease identifier” for decomposition of the event log which resulted in ten decomposed partitions of the event log each containing only activities relevant to specific disease. Subsequently in phase 3, feature extraction was employed over each abstracted and decomposed the event log subset. Ten feature profiles were extracted with each having matrix of features related to different aspects of the event log. The extracted features and their description are detailed in Table II.

In phase 4, the traces clustering step was executed for each log subset using clustering algorithm. For each cluster produced by performing trace clustering, a process model is discovered. For process discovery, the inductive miner algorithm was used which produced a petri net model. Then for each model discovered, the behavioural quality of process models is assessed using well known fitness, precision, and f-score metrices. Subsequently structural quality of each discovered process model is evaluated using inverse arc metric, model density and places count. An exhaustive evaluation was carried out until clustering, model discovery and evaluation of each sublog was achieved. Trace clustering produced several process models, therefore the average values of behavioural and structural quality of process models were used for evaluation of process models. Since model discovery is computationally expensive task, therefore the number of minutes taken to discover final models was used as performance indicator. Finally, results were compared against two baselines using structural complexity measure, behavioural complexity measures and time performance.

We implemented our proposed framework using PM4Py library [56] which is implemented in Python language. In terms of hardware and operating system configuration, we evaluated our solution on Intel(R) Core i7-3.40GHz CPU with 12 GB of RAM and Windows 10 Pro (64-bit) operating system.

5.0. RESULTS

In this section, we present the results obtained by application of the proposed framework for trace clustering. We used two baseline trace clustering techniques Jablonski et al. [20] and Faizan et al. [24] for the experimental evaluation of our

approach and refer to them as baseline 1 and baseline 2 throughout this section.

The experimental evaluation results, as presented in Fig. 3, reveal that in the context of behavioural quality of the process models, our proposed framework increases the precision and f-score of the resulting process models. We obtained 38% and 10% improvement in the precision quality of the resulting process models compared with the two baseline approaches, respectively. Moreover, compared with the two baseline approaches, 40% and 37% percent improvement in the f-score was achieved. Although compared with baseline 2, an improvement in fitness, precision, and f-score of the process models can be observed, however, our approach slightly under-performs in terms of fitness quality against baseline 1.

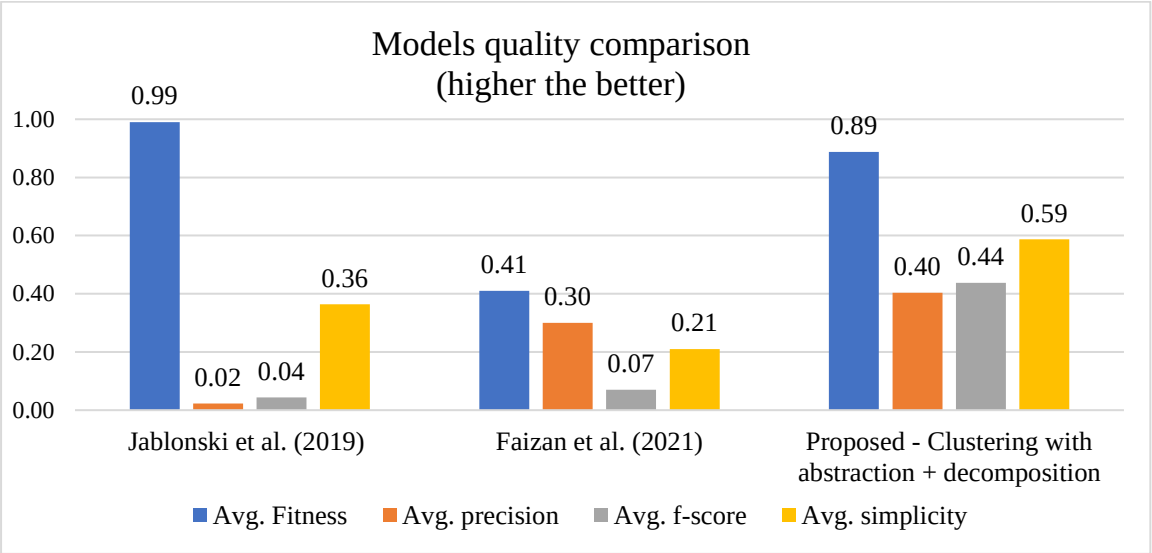

Fig 3. Structural and behavioural quality comparison against baseline

In terms of the complexity of the process models, an increase in simplicity of the process model can be observed as visualized in Fig. 4. Using the proposed framework, we were able to reduce the complexity of the process models by 23% and 38%, respectively in terms of inverse arc metric, as compared with the two baseline approaches. In order to

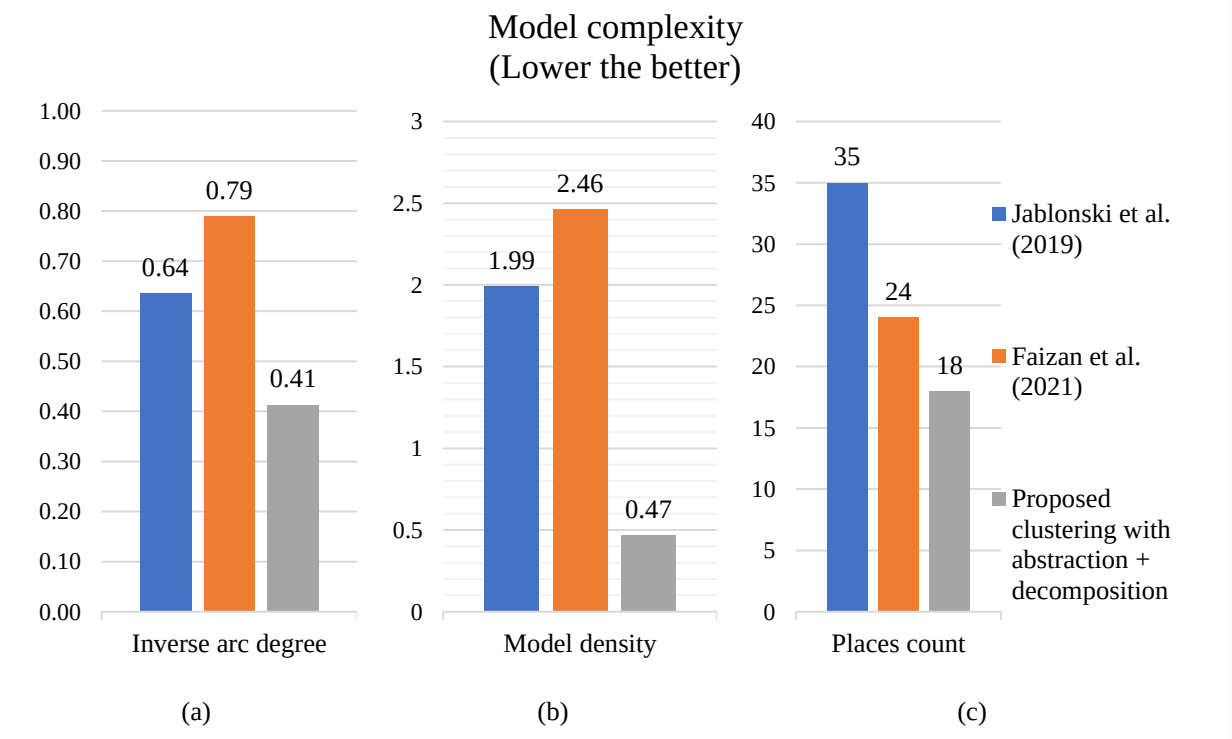

Fig 4. Comparison of model complexity against baseline

present a more complete picture of model complexity reduction, we used two additional structural quality assessment metrics i.e., model density and places count. The density of a model is defined as the ratio between the total number of arcs in the model and the maximum possible number of arcs. The greater the density, the more complex the model of the process. On the other hand, the places count represents number of nodes in the resulting process model. The greater number the nodes in process model, mode complicated it is. Both of these metrices are negatively correlated with understandability of the process models. Therefore, the lower values of these metrices results in more comprehendible process models [14]. In terms of density, our proposed framework produces 76 and 81 percent less simpler models compared with two baselines. Moreover, complexity reduction in terms of places count, we have achieved up to 49 percent of improvement compared with baselines.

A third perspective of the evaluation was the time performance, where the results of our proposed framework were compared against the baseline in terms of overall time taken for trace clustering, model discovery, and model evaluation. Time performance in process mining refers to the ability of the process mining tool to quickly and efficiently analyse large amounts of event data and generate process models.

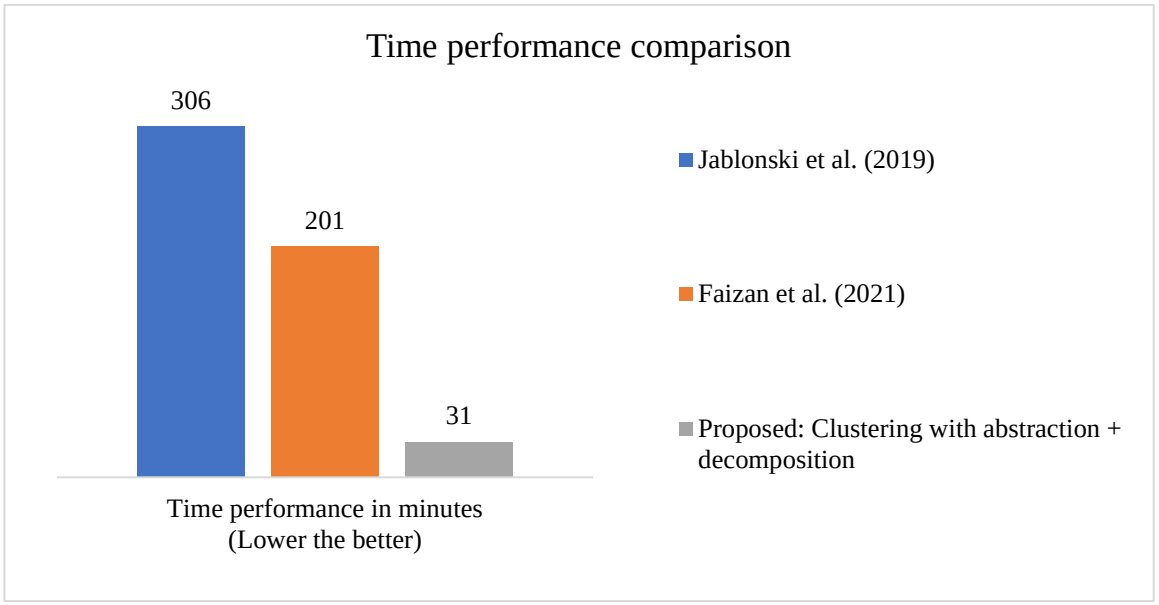

Fig 5. Time performance comparison against baseline

The comparison of time performance against the baseline is shown in Fig. 5. It is evident that the application of our trace clustering framework results in a significant 89 and 85 percent improvement in the time performance of trace clustering and model discovery process against both the baseline works. Despite adding further elements to the existing trace clustering framework, our approach performs much more efficiently than the existing trace clustering framework.

Overall, the experimental results indicate that the proposed farmwork has the potential to enhance both behavioural and structural quality of the process models in addition to enhancing the time performance. In the following section, we will further discuss the significance of these results.

6.0. DISCUSSION

The discussion section aims to provide a comprehensive analysis and interpretation of the experimental results obtained from the proposed framework for managing complexity in process mining. Furthermore, we will discuss the limitations of the proposed framework. The discussion will also explore the practical implications of the findings, highlighting their contribution to the field of process mining and operational processes.

The results of the proposed trace clustering framework show that it improves several key areas of the resulting process models. The enhanced trace clustering framework increases the fitness, precision, and f-score of the process models, indicating a higher quality of the models. At the same time, it leads to a significant increase in the simplicity of the

process models, making them easier to understand and interpret. However, the results also indicate slight underperformance in terms of fitness quality, although the increase in precision and simplicity makes up for this in many cases. Comparing our approach with two baseline trace clustering techniques reveals that the proposed framework results in less complex process models, which are more efficient and less prone to errors. Another critical aspect of the results is the significant improvement in trace clustering and model discovery time performance.

The experimental evaluation results of the proposed framework have significant implications for the field of process mining. Firstly, the improvement in behavioural quality, as demonstrated by the increase in precision and f-score, implies that the process models generated using the proposed framework are of higher quality than those generated using the baseline approaches. This is an important consideration, as high-quality process models are essential for making informed decisions about process improvement.

The reduction in complexity of the process models also has significant implications. With the reduction in complexity, the process models become easier to understand and interpret, making them more accessible to the stakeholders involved in the process. This improved accessibility can lead to greater adoption of process improvement initiatives, as stakeholders are more likely to understand and support changes when they have a clear and concise understanding of the process.

The slightly reduced fitness in the proposed framework compared to baseline 1 implies that the accuracy or the degree to which the process model fits the actual process may have decreased. However, it is important to note that the proposed framework still showed improvement in fitness compared to baseline 2. The slight underperformance observed in terms of fitness quality should be considered in the context of the improvements achieved in precision and f-score. Despite a 10% decrease in fitness relative to baseline 1, substantial increases of up to38% in precision and around 40% in f-score can be observed. The precision of the process models is an indicator of their reliability because it represents the percentage of correctly detected process instances. Therefore, a 38 percent improvement in precision implies a substantial reduction in false positives and better alignment with the actual behaviour of the process. It is widely acknowledged by researchers [57]–[59] that increase in model precision results in a drop of fitness quality. Therefore, in such cases, f-score is used as an alternative model quality indicator which represents a harmonic-mean of precision and fitness quality of process models [14], [17]. Our results already indicate a 40 percent improvement in the f-score, which highlights the overall effectiveness of the proposed framework in capturing the relevant behaviours in the process. Therefore, improvements in f-score compensate for this trade-off. Additionally, the gains in simplicity and the reduction in model complexity further contribute to the overall comprehensibility of the resulting process models. The combination of these factors suggests that the observed 10 percent reduction of fitness is negligible.

The improvement in time performance, as demonstrated by the reduction in the overall time taken for trace clustering and model discovery, also has important implications. In practical applications, it is essential to quickly and efficiently analyse large amounts of event data, and the evaluation results indicate that the proposed framework is capable of doing so. The faster processing against the two baseline approaches can be a significant advantage in many cases, especially in time-sensitive processes where efficient and faster analysis of processes is desired. This efficiency improvement makes the proposed framework a valuable addition to the field of process mining, as it addresses the challenge of handling large amounts of data in a timely manner. Finally, the improvement in time performance can be seen as an indicator of the scalability of our approach. This is particularly important in trace clustering, where multiple models must be discovered and evaluated, making time efficiency gains crucial.

To further enhance the impact of our proposed framework in the field of process mining, there are several future research avenues. The impact of different feature profiles on trace clustering and model evaluation performance can be studied to reduce the number of feature profiles and improve the generalizability of the framework. Moreover, the framework can be applied in different domains, such as production, finance, and manufacturing, to see if it generalizes well and produces accurate and efficient process models in these domains. Additionally, future research can also incorporate human-centric perspectives into proposed framework to capture the impact of incorporating stockholders' feedback in the process modelling to allow a better alignment between the automated analysis and the human aspects of information systems. Finally, usage of behavioural features raises ethical and privacy concerns where users can be identified based on the behavioural patterns they follow. The proposed framework can be integrated with methods to ensure data privacy and anonymization to ensure protection of individuals' privacy rights and promote trust and ethical practice in the field of process mining and information systems.

In conclusion, the experimental evaluation results have important implications for the field of process mining. The improvements in behavioural quality, complexity, and time performance demonstrate that the proposed framework is a valuable addition to the field, as it addresses common challenges faced in analysing event data and generating process models. These results can be leveraged by organizations looking to improve their processes, as it is evident that the proposed framework can lead to high-quality, accessible, and efficient process models.

7.0. CONCLUSION

In this paper, we analysed existing trace clustering framework for dealing with complexity issues in process mining. It was found that existing frameworks work well when the event log has trace-level variation. However, when a fine-granular event log is faced with, the trace clustering alone does not help much in reducing the complexity of the process models and produces less precise process models. In line with these limitations, we presented an improved trace clustering framework with an integration of event log abstraction and decomposition to realize more straightforward, precise, and efficient process models. Evaluation of the proposed framework was performed on a real-world event log dataset. A comparison with existing trace clustering approaches reveals that our proposed framework results in comparatively less complex and more precise process models. Moreover, trace clustering using our proposed framework is much more efficient than the existing framework in the context of time performance. The improved time performance in trace clustering has several implications for the process mining field, especially in the context of mining complex and unstructured processes where timely detection of bottlenecks are crucial. Moreover, an increase in time performance also indicates the scalability of the proposed framework. An interesting research avenue for our future work can be investigating the impact of different feature profiles on trace clustering and model evaluation performance. Such an evaluation will help in reducing number of feature profiles and pave the way for further improving the time performance and the generalizability of proposed framework.

REFERENCES

[1] C. dos Santos Garcia *et al.*, "Process mining techniques and applications – A systematic mapping study," *Expert Syst. Appl.*, vol. 133, 2019, pp. 260–295, doi: 10.1016/j.eswa.2019.05.003.

[2] P. Palangsantikul, P. Porouhan, N. Premchaiswadi, and W. Premchaiswadi, "Emotion analytics with process mining," *Malaysian J. Comput. Sci.*, 2019, pp. 109–131, doi: 10.22452/mjcs.sp2019no2.7.

[3] C. Ortmeier, N. Henningsen, A. Langer, A. Reiswich, A. Karl, and C. Herrmann, "Framework for the integration of Process Mining into Life Cycle Assessment," *Procedia CIRP*, vol. 98, Jan. 2021, pp. 163–168, doi: 10.1016/J.PROCIR.2021.01.024.

[4] H.-J. Cheng and A. Kumar, "Process mining on noisy logs — Can log sanitization help to improve performance?," *Decis. Support Syst.*, vol. 79, 2015, pp. 138–149, doi: 10.1016/j.dss.2015.08.003.

[5] M. Fani Sani, S. J. van Zelst, and W. M. P. van der Aalst, "The impact of biased sampling of event logs on the performance of process discovery," *Computing*, vol. 103, no. 6, Jun. 2021, pp. 1085–1104, doi: 10.1007/s00607-021-00910-4.

[6] Z. Lamghari, M. Radgui, R. Saidi, and M. D. Rahmani, "An operational support approach for Mining Unstructured Business Processes," *Rev. Informática Teórica e Apl.*, vol. 28, no. 1, 2021, pp. 22–38, doi: 10.22456/2175-2745.106277.

[7] S. Ya'acob *et al.*, "Visual analytics evaluation process: Practice guidelines for complex domain," *Malaysian J. Comput. Sci.*, vol. 2019, no. 1, Nov. 2019, pp. 118–134, doi: 10.22452/MJCS.SP2019NO1.9.

[8] C. W. Günther and W. M. P. Van Der Aalst, "Fuzzy mining - Adaptive process simplification based on multi-perspective metrics," in *Lecture Notes in Computer Science*, 2007, vol. 4714 LNCS, pp. 328–343. doi: 10.1007/978-3-540-75183-0_24.

[9] A. F. D. Gomes, A. C. W. G. de Lacerda, and J. R. da Silva Fialho, "Comparative analysis of process mining algorithms in Python," in *Smart Objects and Technologies for Social Good*, 2021, vol. 401 LNICST, pp. 27–43. doi: 10.1007/978-3-030-91421-9_3.

[10] M. Imran, S. Hamid, and M. A. Ismail, "Advancing Process Audits With Process Mining: A Systematic Review of Trends, Challenges, and Opportunities," *IEEE Access*, vol. 11, 2023, pp. 68340–68357, doi: 10.1109/ACCESS.2023.3292117.

[11] J. Mendling, H. A. Reijers, and J. Cardoso, "What makes process models understandable?," in *Business Process Management*, 2007, vol. 4714 LNCS, pp. 48–63. doi: 10.1007/978-3-540-75183-0_4.

[12] H. A. Reijers and J. Mendling, "A study into the factors that influence the understandability of business process models," *IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans*, vol. 41, no. 3, May 2011, pp. 449–462, doi: 10.1109/TSMCA.2010.2087017.

[13] C. Li, S. J. van Zelst, and W. M. P. van der Aalst, "An Activity Instance Based Hierarchical Framework for Event Abstraction," in *2021 3rd International Conference on Process Mining (ICPM)*, Oct. 2021, pp. 160–167. doi: 10.1109/ICPM53251.2021.9576868.

[14] M. Imran, M. A. Ismail, S. Hamid, and M. H. N. M. Nasir, "Complex Process Modeling in Process Mining: A Systematic Review," *IEEE Access*, vol. 10, 2022, pp. 101515–101536, doi: 10.1109/ACCESS.2022.3208231.

[15] A. Kumaraguru, "A Framework for Unsupervised Event Abstraction using Hidden Markov Models Master's Thesis," RWTH Aachen University, 2021. [Online]. Available: https://www.pads.rwth-aachen.de/cms/PADS/Studium/Abgeschlossene-Abschlussarbeiten/2021/~qhkcx/A-Framework-for-Unsupervised-Event-Abstr/

[16] C. Tsagkani and A. Tsalgatidou, "Process model abstraction for rapid comprehension of complex business processes," *Inf. Syst.*, vol. 103, 2022, p. 101818, doi: 10.1016/j.is.2021.101818.

[17] R. Conforti, M. La Rosa, and A. H. M. ter Hofstede, "Filtering Out Infrequent Behavior from Business Process Event Logs," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 2, Feb. 2017, pp. 300–314, doi: 10.1109/TKDE.2016.2614680.

[18] N. Tax, N. Sidorova, and W. M. P. Van Der Aalst, "Discovering more precise process models from event logs by filtering out chaotic activities," *J. Intell. Inf. Syst.*, vol. 52, no. 1, 2019, pp. 107–139, doi: 10.1007/s10844-018-0507-6.

[19] N. Tax, N. Sidorova, W. M. P. van der Aalst, and R. Haakma, "LocalProcessModelDiscovery: Bringing Petri Nets to the Pattern Mining World," in *Proceedings of International Conference on Applications and Theory of Petri Nets and Concurrency*, Jun. 2018, vol. 10877 LNCS, pp. 374–384. doi: 10.1007/978-3-319-91268-4_20.

[20] S. Jablonski, M. Röglinger, S. Schönig, and K. Wyrtki, "Multi-Perspective Clustering of Process Execution Traces," *Enterp. Model. Inf. Syst. Archit.*, vol. 14, no. 2, 2019, pp. 1–22, doi: 10.18417/emisa.14.2.

[21] W. M. P. van der Aalst, "Process discovery from event data: Relating models and logs through abstractions," *WIREs Data Min. Knowl. Discov.*, vol. 8, no. 3, May 2018, p. e1244, doi: 10.1002/widm.1244.

[22] A. Thaduri, S. M. Famurewa, A. K. Verma, and U. Kumar, "Process Mining for Maintenance Decision Support," in *System Performance and Management Analytics*, Springer Singapore, 2019, pp. 279–293. doi: 10.1007/978-981-10-7323-6_23.

[23] H. R'Bigui and C. Cho, "The state-of-the-art of business process mining challenges," *Int. J. Bus. Process Integr. Manag.*, vol. 8, no. 4, 2017, pp. 285–303, doi: 10.1504/IJBPIM.2017.10009731.

[24] M. Faizan, M. F. Zuhairi, and S. Ismail, "Process discovery enhancement with trace clustering and profiling," *Ann. Emerg. Technol. Comput.*, vol. 5, no. 4, 2021, pp. 1–13, doi: 10.33166/AETIC.2021.04.001.

[25] M. Siek and R. M. G. Mukti, "Business process mining from e-commerce event web logs: Conformance checking and bottleneck identification," in *2nd International Conference on Biospheric Harmony Advanced Research*, Apr. 2021, vol. 729, no. 1, p. 12133. doi: 10.1088/1755-1315/729/1/012133.

[26] W. M. P. Van Der Aalst *et al.*, *Process Mining Handbook*, vol. 448. Cham: Springer International Publishing, 2022. doi: 10.1007/978-3-031-08848-3.

[27] A. Kaouni, G. Theodoropoulou, A. Bousdekis, A. Voulodimos, and G. Miaoulis, "Visual analytics in process mining for supporting business process improvement," *Front. Artif. Intell. Appl.*, vol. 338, 2021, pp. V–VI, doi: 10.3233/FAIA210089.

[28] S. J. van Zelst, F. Mannhardt, M. de Leoni, and A. Koschmider, "Event abstraction in process mining: literature review and taxonomy," *Granul. Comput.*, vol. 6, no. 3, Jul. 2021, pp. 719–736, doi: 10.1007/s41066-020-00226-2.

[29] D. Chapela-Campa, M. Mucientes, and M. Lama, "Simplification of Complex Process Models by Abstracting Infrequent Behaviour," in *Service-Oriented Computing*, Oct. 2019, vol. 11895 LNCS, pp. 415–430. doi: 10.1007/978-3-030-33702-5_32.

[30] M. Acheli, D. Grigori, and M. Weidlich, "Efficient Discovery of Compact Maximal Behavioral Patterns from Event Logs," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11483 LNCS, Springer Verlag, 2019, pp. 579–594. doi: 10.1007/978-3-030-21290-2_36.

[31] P. Delias, M. Doumpos, E. Grigoroudis, and N. Matsatsinis, "A non-compensatory approach for trace clustering," *Int. Trans. Oper. Res.*, vol. 26, no. 5, Sep. 2019, pp. 1828–1846, doi: 10.1111/itor.12395.

[32] M. de Leoni and S. Dündar, "Event-log abstraction using batch session identification and clustering," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, Mar. 2020, pp. 36–44. doi: 10.1145/3341105.3373861.

[33] A. Cuzzocrea, F. Folino, M. Guarascio, and L. Pontieri, "Deviance-Aware Discovery of High-Quality Process Models," *Int. J. Artif. Intell. Tools*, vol. 27, no. 07, 2018, p. 1860009, doi: 10.1142/S0218213018600096.

[34] P. Wang, W. Tan, A. Tang, and K. Hu, "A Novel Trace Clustering Technique Based on Constrained Trace

Alignment," in *Proceedings of International Conference on Human Centered Computing*, 2018, pp. 53–63. doi: 10.1007/978-3-319-74521-3_7.
[35] J. Evermann, T. Thaler, and P. Fettke, "Clustering traces using sequence alignment," in *Proceedings of International Conference on Business Process Management*, 2016, vol. 256, pp. 179–190. doi: 10.1007/978-3-319-42887-1_15.
[36] C. W. Günther, A. Rozinat, and W. M. P. Van Der Aalst, "Activity Mining by Global Trace Segmentation," in *International Conference on Business Process Management*, 2010, pp. 128–139. doi: 10.1007/978-3-642-12186-9_13.
[37] G. M. Veiga and D. R. Ferreira, "Understanding Spaghetti Models with Sequence Clustering for ProM," in *International Conference on Business Process Management*, 2010, vol. 43 LNBIP, pp. 92–103. doi: 10.1007/978-3-642-12186-9_10.
[38] P. Delias, M. Doumpos, E. Grigoroudis, P. Manolitzas, and N. Matsatsinis, "Supporting healthcare management decisions via robust clustering of event logs," *Knowledge-Based Syst.*, vol. 84, Aug. 2015, pp. 203–213, doi: 10.1016/j.knosys.2015.04.012.
[39] N. Assy, B. F. van Dongen, and W. M. P. van der Aalst, "Discovering Hierarchical Consolidated Models from Process Families," in *Lecture Notes in Computer Science*, vol. 10253 LNCS, Springer, Cham, 2017, pp. 314–329. doi: 10.1007/978-3-319-59536-8_20.
[40] F. Mannhardt and N. Tax, "Unsupervised Event Abstraction using Pattern Abstraction and Local Process Models," in *29th International Conference on Advanced Information Systems Engineering*, 2017, pp. 55–63.
[41] N. Wang, S. Sun, Y. Liu, and S. Zhang, "Business Process Model Abstraction Based on Fuzzy Clustering Analysis," *Int. J. Coop. Inf. Syst.*, vol. 28, no. 03, Sep. 2019, p. 1950007, doi: 10.1142/S0218843019500072.
[42] Y. Sun and B. Bauer, "A Novel Top-Down Approach for Clustering Traces," in *Proceedings of International Conference on Advanced Information Systems Engineering*, 2015, vol. 9097, pp. 331–345. doi: 10.1007/978-3-319-19069-3_21.
[43] S. J. Van Zelst and Y. Cao, "A Generic Framework for Attribute-Driven Hierarchical Trace Clustering," in *International Conference on Business Process Management*, 2020, pp. 308–320.
[44] R. P. J. C. Bose and W. M. P. Van Der Aalst, "Trace Clustering Based on Conserved Patterns: Towards Achieving Better Process Models," in *Lecture Notes in Business Information Processing*, vol. 43 LNBIP, Springer, Berlin, Heidelberg, 2009, pp. 170–181. doi: 10.1007/978-3-642-12186-9_16.
[45] B. Hompes, J. Buijs, W. Aalst, P. Dixit, and J. Buurman, "Discovering Deviating Cases and Process Variants Using Trace Clustering," in *Benelux Conference on Artificial Intelligence (BNAIC 2015)*, 2015, p. 8.
[46] J. De Weerdt, S. vanden Broucke, J. Vanthienen, and B. Baesens, "Active Trace Clustering for Improved Process Discovery," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 12, 2013, pp. 2708–2720, doi: 10.1109/TKDE.2013.64.
[47] A. Rozinat, A. K. A. de Medeiros, C. W. Günther, A. J. M. M. Weijters, and W. M. P. van der Aalst, "The Need for a Process Mining Evaluation Framework in Research and Practice," in *Business Process Management Workshops*, Springer Berlin Heidelberg, 2008, pp. 84–89. [Online]. Available: https://doi.org/10.1007%2F978-3-540-78238-4_10
[48] T. Thaler, S. Ternis, P. Fettke, and P. Loos, "A Comparative Analysis of Process Instance Cluster Techniques," *Proc. der 12. Int. Tagung Wirtschaftsinformatik (WI 2015)*, 2015, pp. 423–437.
[49] X. Lu, S. A. Tabatabaei, M. Hoogendoorn, and H. A. Reijers, "Trace Clustering on Very Large Event Data in Healthcare Using Frequent Sequence Patterns," in *Lecture Notes in Computer Science*, vol. 11675 LNCS, Springer Verlag, 2019, pp. 198–215. doi: 10.1007/978-3-030-26619-6_14.
[50] M. Song, C. W. Gunther, and W. M. P. van der Aalst, "Trace Clustering in Process Mining," in *International Conference on Business Process Management*, 2009, vol. 17, pp. 109–120.
[51] M. Song, H. Yang, S. H. Siadat, and M. Pechenizkiy, "A comparative study of dimensionality reduction techniques to enhance trace clustering performances," *Expert Syst. Appl.*, vol. 40, no. 9, 2013, pp. 3722–3737, doi: 10.1016/j.eswa.2012.12.078.
[52] H. Li, "Trace Clustering : A Preprocessing Method to Improve the Performance of Process Discovery," *Int. J. Sci. Eng. Appl.*, vol. 10, no. 09, 2021, pp. 116–121.
[53] N. Tax, B. Dalmas, N. Sidorova, W. M. P. van der Aalst, and S. Norre, "Interest-driven discovery of local process models," *Inf. Syst.*, vol. 77, Sep. 2018, pp. 105–117, doi: 10.1016/J.IS.2018.04.006.
[54] B. N. Yahya, M. Song, H. Bae, S. Sul, and J.-Z. Wu, "Domain-driven actionable process model discovery," *Comput. Ind. Eng.*, vol. 99, 2016, pp. 382–400, doi: 10.1016/j.cie.2016.05.010.
[55] B. F. van Dongen, "Real-life event logs - Hospital log," *4TU Research Data Consortium*, 2011. https://data.4tu.nl/articles/dataset/Real-life_event_logs_-_Hospital_log/12716513/1 (accessed Jan. 12, 2023).
[56] A. Berti, van Z. Sebastiaan, and W. van der Aalst, "Process Mining for Python (pm4py): Bridging the Gap between Process-and Data Science," in *Proceedings of the ICPM Demo Track 2019, co-located with 1st International Conference on Process Mining (ICPM 2019)*, 2019.

[57] E. Benevento, P. M. Dixit, M. F. Sani, D. Aloini, and W. M. P. van der Aalst, “Evaluating the Effectiveness of Interactive Process Discovery in Healthcare: A Case Study,” in *Lecture Notes in Business Information Processing*, vol. 362 LNBIP, Springer, Cham, 2019, pp. 508–519. doi: 10.1007/978-3-030-37453-2_41.

[58] D. Chapela-Campa, M. Mucientes, and M. Lama, “Understanding complex process models by abstracting infrequent behavior,” *Futur. Gener. Comput. Syst.*, vol. 113, 2020, pp. 428–440, doi: 10.1016/j.future.2020.07.030.

[59] Y. Huang, L. Zhong, and Y. Chen, “Filtering Infrequent Behavior in Business Process Discovery by Using the Minimum Expectation,” *Int. J. Cogn. Informatics Nat. Intell.*, vol. 14, no. 2, Apr. 2020, pp. 1–15, doi: 10.4018/IJCINI.2020040101.