vAMoS: eVent Abstraction via Motifs Search

Gemma Di Federico and Andrea Burattin

Technical University of Denmark, Kgs. Lyngby, Denmark
gdfe@dtu.dk

Abstract. Process mining analyzes events that are logged during the execution of a business process. The level of abstraction at which a process model is recorded is reflected in the level of granularity of the data in the event log. When process activities are recorded as sensors readings, typically, they are very fined-grained and therefore difficult to interpret. To increase the understandability of the process model, events need to be abstracted into higher-level activities. This paper proposes vAMoS, a trace-based approach for event abstraction, which focuses on the identification of motifs on the traces, allowing some level of flexibility. The objective is the identification of recurring motifs on the traces in the event log. The presented algorithm uses a distance function to deal with the variability in the execution of activities. The result is a set of readable and interpretable motifs.

Keywords: Event abstraction · Motifs search · Sensor data.

1 Introduction

Process mining [1] analyzes events that are logged during the execution of a business process. These events contain information on the activity executed, the resources involved, the execution time, etc. Control-flow discovery is the task of generating a process model that describes the process executions as collected in the logged data. The degree of abstraction to which the process model is represented, is reflected in the level of granularity of the recorded data. When data in the event log are recorded in a too fine grained fashion, the process model becomes too complex and no longer interpretable, thus less effective in extracting knowledge regarding of the underlying process.

When Business Process Management is used with data coming from Internet of Things [6], assuming a one-to-one mapping between an event in the log and a sensor reading, typically leads to too-fine grained event logs. Therefore, these is a need to abstract events such that they are able to represent more meaningful higher-level activity that can be related to a process level. To accomplish this task, a certain degree of expertise and knowledge may be required to be able to properly configure abstraction mechanisms.

When the data under investigation refers to human activities, human behavior and processes, additional challenges must be faced. Human related processes, such as daily personal routines, are unstructured and characterized by variability [5]. Consequently, it is difficult to define the set of activities carried out and their execution order. Additionally, when human behavior is collected in form of sensors data, the two challenges coalesce. The result is that, on the one hand, activities are expressed in the form of too fine-grained sensors readings; on the other hand, there is complexity in aggregating the activities carried out and understand what these express.

This problem has received substantial interest in the literature. However, the solutions available in the literature do not address the two sides of the problem together. Our research aims at finding a better solution for this challenging problem, by leveraging solutions devised in different scientific disciplines, i.e., biology. The method we propose is called vAMoS and it aims at recognizing common sequences of activities that show a high-level of similarity among traces. Therefore, the research conducted aims to address the following two questions:

RQ1 Is the technique able to identify meaningful patterns that can be abstracted as higher-level activities?
RQ2 Is the technique able to abstract behaviors from an event log by aggregating patterns?

vAMoS focuses on the identification of patterns of behavior at the trace level, in the form of sub-traces which show high-level of commonalities. The algorithm produces the list of patterns that can be used to abstract the original event log. To exemplify the problem as well as the devised solution better, let's introduce an example referring to the morning routine of a person who lives in an environment equipped with sensors.

Example 1. Every morning the person gets up form the bed area (🛏), eats some food (🍎) and drinks a coffee (☕). Rarely, after waking up, the person goes directly to the work (🚲). We can consider the following event log:

$t_1 = \langle$🛏, 🍎, 🍎, ☕$\rangle$ $t_2 = \langle$🛏, 🍎, ☕, ☕$\rangle$ $t_3 = \langle$🛏, 🚲, 🚲, 🚲, 🚲$\rangle$

The typical behavior explained before (getting up and having breakfast, either as coffee or eating something) is indeed reflected in the traces, even though there is no single sub-sequence capturing it properly.

The objective is therefore to recognize the series of activities that are common in most traces. Each recognized sequence must be expressive enough to be explicable in the form of a higher-level activity.

The rest of the paper is structured as follows. Sect. 2 presents the related work. The solution is presented in Sect. 3 that is then evaluated in Sect. 4 including a discussion of the results, and the limitation of the approach. Sect. 5 concludes the paper.

2 Related Work

The problem of event abstraction has been tackled by several papers presenting many techniques, ranging from supervised to unsupervised approaches [15].

The majority of event abstraction algorithms require a knowledge expert to take active role in the analysis. Experts' knowledge is crucial to obtain information on the groups of sensors involved in the execution of a specific activity, to identify the main activities, and for the evaluation and verification of the results. However, there are two drawbacks that limit the expert's participation in the recognition of activities. As previously introduced, human processes are typically flexible and variable, therefore it is challenging for the expert to define exact rules for their recognition, in particular considering the fine granularity of the data collected. Another drawback concerns the set of recognizable activities, limited to the one provided by the expert. Instead, a "bottom-up approach", where possible activities are elicited directly from the data, allows the discovery of activities that were not foreseen beforehand. Consequently, the abstraction identifies a larger set of activities and is more pertaining to the reality observed in the log. All things considered, experts' knowledge should only have a supportive role and should not be considered as the main driver of the abstraction. For these reasons researchers investigated alternative ways to initialize the abstraction. In the literature, techniques for events abstraction can be divided in model- or trace-based. The former aims to derive process models that represent activities, the latter focuses on the identification of activities at a trace-level.

Model-based approaches aim to derive a model for each recognized activity in the event log. Leotta et al. [8] propose a model-based approach that consider the log as a "complex trajectory", and their objective is to identify sub-trajectories that are portions of the log referring to the same action. The algorithm identifies very short paths, hence the granularity of the abstraction remains too fine. Mannhardt et al. [9] proposed a supervised event abstraction method that makes use of behavioral patterns. Given a set of activities pattern, they build an abstracted model. After that, alignment techniques [3] are used to map low-level events in the event log to activities in the abstraction model. The approach is extended by the use of Local Process Model [13] (LPM) to describe the set of patterns: each LPM represents a high-level activity. The set of LPMs is then filtered and used with the abstraction method proposed in the previous version. LPMs can suffer from the over-generalization problem: the same LPM could be matched by potentially infinite different sequences of low-level events. What is more, LPMs describe small patterns in the log, composed by only three to five transitions. Another model-based abstraction solution is proposed by Baier et al. [2], which exploits imperative process model to derive constraints, following the same line of reasoning as the approach above. The proposed technique requires as input a process model describing the behavior as a set of activities, as well as a low-level event log. The objective is to map activities in the model with events in the log. The approach extracts declarative constraints from both the model and the log to build matching constrains to reduce the number of possible mappings. The mapping is also reduced using natural language processing, with a matching based on activities' labels and external knowledge. It follows from the above that a rigid and well-defined idea of the expected model is required, limiting the recognition to a known set of activities represented in the process model.

Trace-based approaches, on the other hand, focus on the identification of commonalities between traces. In [7], de Leoni et al. propose a technique that aims to divide traces into batch sessions. Each trace is seen as a sequence of sessions of events. The sessions are converted into vectors which abstract the behavior observed inside the sessions, and then are clustered. The centroids are used for naming the activities. Each session is abstracted as one high-level activity execution, and K-Means is one of the proposed techniques for the clustering step. The main drawback lies in the choice of the session threshold parameter used for trimming sessions: this could have potentially impactful effects further in the analysis since activities could be missed or wrongly identified. The use of clustering algorithms, such as K-Means, is common among abstraction approaches proposed in the literature. Van Eck et al. [14] addressed the challenges of mapping sensor measurements to human activities, and grouping activities into process instances. The approach starts with the segmentation of sensor data in windows to be labeled. Relevant features are calculated for each of them, and subsequently the segments are clustered using K-Means. At this point, a domain expert assigns labels to the clusters, and segments can be grouped together to create activities. This technique was applied on a process regarding only one smart product equipped with sensors, so the performances in a real environment have not been evaluated. A limitation of the approaches that make use of K-means concern the impacts caused by the choice of the number of clusters (K). What is more, when number of low-level activities is large, the algorithm generates sparse clustering points which compromises the quality of the clusters.

When broadening the scope of the work to other domains, similarities can be found in bioinformatics and with the problem of protein sequence classification. The authors in [11] apply the sequential K-Means algorithm to sequences of data. Their goal is to find clusters that represent proteins, but considering the accumulation of mutations. To bring their problem into this paper's domain, each sequence can be seen as a trace, and the proteins represent the activities to be recognized (including mutations, i.e. noise). The algorithm provides as output clusters, that are activities identified, in form of sequences of events.

Regardless of the progress in the area, the major problem of event abstraction persists. In particular, there is no predominant trend between model and trace-based approaches. Although several studies have indicated good performances, little attention has been given to the granularity of the data source. In fact, only a few works in the literature demonstrate to perform well with low-level sensors data, e.g. [10]. For this reason, in the rest of this paper we use this work as a means of comparison to our algorithm.

3 Solution

The approach proposed in this paper aims at finding sequences of recurring activities over the event log that, when identified on a low-level log (e.g., where activities refer to sensor data), can be interpreted as higher-level activities. The contribution of this work is called vAMoS, a motif search algorithm able to deal with recurring and variable patterns, that focuses only on observed sequences. vAMoS is implemented as a Java application, and the code is freely available[1].

The origin of vAMoS can be traced to the qPMS algorithm [12], a quorum Planted Motif Search approach designed to identify patterns (called *motifs*) in biological sequences of proteins, such that these motifs occur in most sequences provided as input. The objective of vAMoS is finding motifs but providing a more complex and accurate mechanism for their identification (e.g., to accept slight variations in how the motif is actually observed). To establish parallelism with the concept of *alignment* between traces [3]: in alignments, given two traces it is possible to calculate their cost (i.e., the extent of their similarity); in vAMoS, given two traces and a maximum cost it is possible to identify common parts of the traces that are similar (up to the cost).

Intuitively, a motif is a recurring sequence of activities that can be recognized in the log. To identify motifs, vAMoS first constructs a set of *candidate* motifs which are then verified on the traces of the event log. The verification procedure checks whether the motif is observed, up to a certain dissimilarity, in a minimum number of traces. The length of the motif is decided by the user. The construction of the set of candidate motifs could involve the construction of all possible sequences of activities but this would be unfeasible, especially since most of the candidates would be completely irrelevant (being very different from any trace on the event log). However, since we are interested in all possible motifs, even those that are not perfectly matched in the log, it is not enough to just consider all possible sub-traces of the log. Therefore, we opted for a two-step approach: first, identify the sub-traces that define the "alphabet" of our candidate motifs, and then combine these sub-traces in order to populate our set of candidate motifs. With this approach, the alphabet would be based on actual observations coming from the log, but the way candidate motifs are constructed allows for previously unseen candidates.

The first step consists of constructing the alphabet of sub-traces. Since the objective is to consider only observed sequences, we have to derive all the sequences, inside the traces, which are of a given length defined by the user. Hence, for each trace in the log, all the sub-traces of the given length are stored (like a sliding window). A sub-trace is called *n-gram*, while the set of all the sub-traces identified is named *n-set*. Considering Example 1 and setting 2 as the n-grams length, given the traces t_1, t_2, t_3, the n-set is composed by all the pairs observed in the log: $N(E_L, 2) = \{$⟨🛏, 🍎⟩, ⟨🍎, 🍎⟩, ⟨🍎, ☕⟩, ⟨☕, ☕⟩, ⟨🛏, 🚲⟩, ⟨🚲, 🚲⟩$\}$.

The second step consists of the construction of the actual set of candidate motifs, meaning that all the n-grams have to be combined in order to reach the desired motif length. To construct the set of candidate motifs, starting from the n-set, we concatenate all the elements in the n-set between each other, up to reach the desired motif length (under approximation, in case the motif length is not multiple of the n-gram length). The obtained set comprises all possible combina-

[1] The source code can be found at https://dx.doi.org/10.5281/zenodo.6378497.

tions. Continuing with Example 1, considering an n-gram length $l_n = 2$ and motif length $l_m = 4$, the set of candidate motifs is the product between each element of the n-set, $C(E_L, l_n, l_m) = \{\langle$🛏, 🍎, 🛏, 🍎$\rangle, \langle$🛏, 🍎, 🍎, 🍎$\rangle, \langle$🛏, 🍎, 🍎, ☕$\rangle, \ldots\}$. It becomes quickly clear that not all candidate motifs are meaningful.

Once the set of candidate motifs is available, the next step is the verification of each of its elements in order to establish whether it is a motif indeed. The verification requires the concepts of *cost*, *distance* and *quorum*. The objective is to check if there are enough traces containing a given motif, within a certain similarity. The concept of similarity, in the context of this paper, establishes whether two activities can be considered equivalent w.r.t. the motif they belong. The cost relation indicates the extent of the interchangeability of two activities. E.g., given two activities a_1, a_2, if the $cost(a_1, a_2)$ is close to 0, then a_1 and a_2 are interchangeable (i.e., they are equivalent); if the value is close to 1 it means they are not. The *cost* relation is expected to be provided by the user of the approach, since it is heavily domain dependent. For example, if we are considering two completely unrelated activities, and we have enough knowledge to state that they cannot be performed in the same context, therefore they cannot be interchanged, we will define their cost as a value close to 1. On the opposite, if we have two activities that are commutable, we define their cost as zero. Considering the IoT setting, this functionality can be useful to define relations between sensors placed on the same object, or sensors that are always activated together.

Given the cost between pairs of activities, we can define the distance d between two traces as the sum of the costs for each pair of activities of the respective traces. The distance function is useful to quantify the variability of non-interchangeable activities in traces which, in turn, is relevant since the same motif may appear multiple times in the log, but each instantiation can have a slightly different configuration. Considering Example 1, it can be noticed that both t_1 and t_2 follows a similar sequence of activities, i.e., moving from the bed area, to the kitchen for having some food and coffee. However, the traces are not exactly equivalent. In particular, if we consider the cost relation $cost = \{($🍎, ☕$, 0.7), ($🍎, 🚲$, 1), ($☕, 🚲$, 1)\}$ then $dist(t_1, t_2) = 0.7$, $dist(t_1, t_3[1, 4]) = 3$, $dist(t_2, t_3[1, 4]) = 3$. Employing a maximum distance threshold d, we would deem t_1 and t_2 within distance; whereas t_1 and t_2 against the sub-trace $t_3[1, 4]$ are not.

As discussed, a motif should be very common in the log to be considered relevant. vAMoS leverages the notion of *quorum* which represents the minimum percentage of traces where the candidate motif should appear (within a certain distance) to be considered relevant. Intuitively, from Example 1, considering the candidate motif $m = \langle$🛏, 🍎, ☕, ☕$\rangle$, with max distance 1, a quorum 100% would not qualify m as a valid motif; whereas a quorum of 60% would. All things considered, we need the ability to check if a candidate motif is contained, within a certain distance d and *cost*, in trace t. The result of the evaluation is a so called motif or verified motif. To check if a candidate motif is a valid motif, we calculate the set of all traces for which at least one sub-trace of it (with the same length as the motif) has a distance less than d against the motif. If there are enough traces (i.e., more than the quorum asks for), then the candidate motif is indeed a motif. The set of all verified candidate motifs is called *Set of motifs*. In particular, the procedure for the verification of a candidate motif starts by computing the number of traces where it appears by checking each possible sub-trace (with the same length as the candidate motif) of each trace in the log. If one sub-trace contains the candidate motif (i.e., its distance is less than a parameter), the respective trace counts as valid, and the number of traces that contains the motif under evaluation is incremented. If the number of traces containing the candidate motif satisfies the quorum, then the candidate motif becomes verified and hence it is returned as a motif. It is worth mentioning that while the set of motifs requires a very specific configuration both in terms of the motif and n-gram lengths, the motifs can be iteratively computed for any configuration and then combined together.

Once the motifs are identified, these can be abstracted. Each motif should be labeled with an activity name, and replaced in the event log. In order to replace a verified motif, it is enough to look for all instances of the motif in each trace (up to a given distance, considering a cost relation) and replace each of them with the abstracted activity. Considering one final time Example 1, we can compute the motifs that fulfill the requirements considering $l_n = 2$, $l_m = 4$, $d = 0.7$, $q = 60\%$, and $cost = \{($🍎, ☕$, 0.7), ($🍎, 🚲$, 1), ($☕, 🚲$, 1)\}$. The set of motifs identified is $M = \{\langle$🛏, 🍎, ☕, ☕$\rangle, \langle$🛏, 🍎, 🍎, ☕$\rangle\}$. These sequences of events could then be abstracted into the higher level activity of "getting ready in the morning".

The motifs computed by vAMoS are saved in form of traces in an XES file. The abstraction requires the set of motifs identified and the original event log. The approach implemented also offers the possibility to abstract the event log, as introduced above, by replacing the motifs in the log with a label. The abstracted event log is returned as an XES file, suitable to be used in process mining analysis.

4 Evaluation

The objective of the evaluation is to verify whether the algorithm is able to fulfill and answer the research questions presented in Section 1. To do so, we conducted tests on real data, for a qualitative assessment. All the evaluations are made by comparing our solution against one other approach in the literature, that is the event abstraction using LPMs [10] (referred to as LPM).

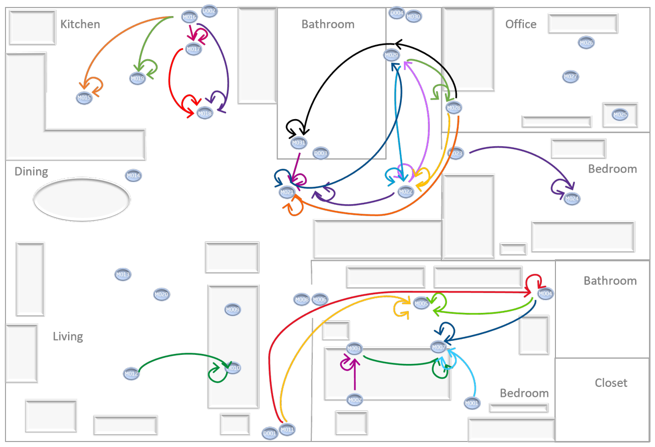
4.1 Evaluation on real data

For the analysis of real data, we opted for a qualitative assessment of the two approaches: vAMoS and LPM. We select the CASAS dataset [4] for this task. The dataset contains sensor data collected in a smart apartment, where a single person is living. It consists of 95 000 events (generated starting from 39 sensors) over 15 cases. The log is labeled with high-level activities, but no information is given on how and when the activities are performed. Therefore, we decided to not consider the actual labeling and just use the real raw data for our experiment. As the resident did not have guidelines to follow, it cannot be assumed that the behaviors were recurrent. To make the dataset homogeneous, we select only cases in which the same set of activities is performed. In addition, the recording interval is reduced keeping only events from 07:00 to 12:00. Sensors not necessary for the recognition of activities (e.g., thermometers and `OFF` values) were filtered out. The resulting log consists of 7 cases, 10 000 events, and 31 sensors involved.

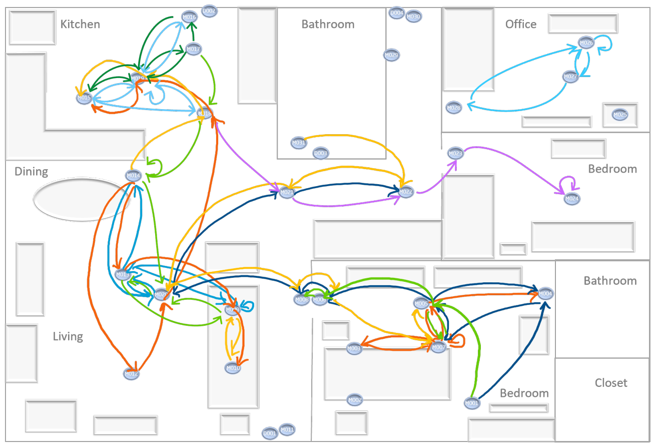
The event log is used as input for the two abstraction algorithms. An iterative approach was followed for vAMoS, starting by searching long motifs and then gradually reducing the length when the algorithm was no longer able to identify verified motifs. A total of 18 motifs are identified. On the other hand, several iterations were necessary to refine the parameter configuration of LPM. To give the algorithm the possibility to discover a wide range of results, LPM was set to produce 30 models, with five transitions and no duplicate activities. All the patterns are represented in form of process models. Two evaluations are conducted, one compares the patterns identified by both the algorithms, while the second evaluation focuses on the analysis of the abstracted event log obtained by the application of vAMoS.

The objective of the first evaluation is to compare the single patterns identified by the two approaches. The assessment is presented in Figure 1 as a visual comparison between the models obtained by the application of the two approaches, projected on top of the floor map provided by CASAS. Even if the objective of LPM is to represent small patterns, between three and five activity nodes (i.e., transitions), the majority of the identified patterns are represented by the alternation between two sensors (see Fig. 1a), thereby these sequences are not very effective at recognizing meaningful higher level activities. On the contrary, results from vAMoS are shown in Fig. 1b, where the patterns are significantly longer and can be interpreted as activities. For instance, the path in the Office room comprises entering the room, moving steadily around the two sensors and then going out; another example (colored in lilac) illustrates the motif of moving from the Kitchen to the Bedroom.

The second evaluation is a comparison between the original log and the abstracted log obtained by the application of vAMoS. After the identification of the motifs, these were replaced in the original log to obtain a new abstract event log. The scarf-plot in Table 1 shows, for each case, a pair of bars: the bar on top represents the raw data from CASAS, the bottom bar presents the data after the processing with vAMoS. Each bar is composed of a series of colors, which indicate different activities (i.e. sensor triggers on top, sensor triggers and patterns on the bottom). Each sensor is associated with a different color, while the identified motifs are all represented by the yellow regions. The width of each colored bar is given by the duration of the activity. The last two columns in Table 1 refer to the number of events in the case replaced by a motif both in terms of duration and events abstracted. The case C5 is an example of the performance



(a) Patterns identified using LPM



(b) Patterns identified using vAMoS

Fig. 1: Floormap and patterns identified by the two approaches evaluated

of the algorithm. As can be noticed from the table, the 66% of the duration of the case is grouped due to a motif. In total, 55% of the duration of the original log is replaced with pattern labels. Concerning events, 45% of the total number of events in the log has been grouped in motifs. Following the same example of C5, 52% of events are abstracted. In case C4 instead, the percentages of the reduction are much lower, meaning that the motifs recognized in this trace are short and with low frequency. It is important to mention that we decided to stop the iterations of vAMoS after obtaining 25 patterns, otherwise we would have a percentage of the replaced log even higher.

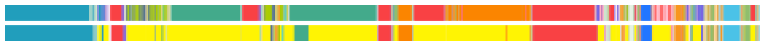
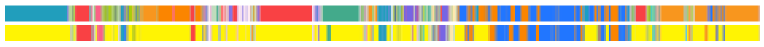
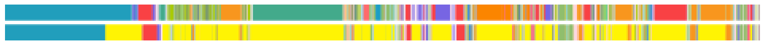
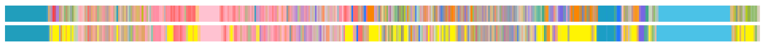
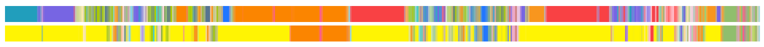
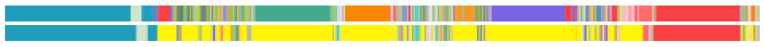
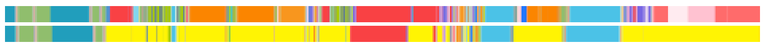
Case	Scarf-plot	Dur	Evs
C1		55%	55%
C2		63%	44%
C3		60%	47%
C4		24%	21%
C5		66%	52%
C6		52%	51%
C7		60%	62%
	Average:	**54%**	**47%**

Table 1: CASAS dataset before and after abstraction

4.2 Discussion and Limitations

The algorithm presented in this paper can be used to abstract activities recorded in fine-grained event logs, for example when events refer to sensors readings. The algorithm was compared to a model-based approach, LPM. The evaluation performed on the real dataset was used to verify whether the algorithm is able to recognize recurrent pattern, that are meaningful, in order to abstract an event log.

The first evaluation conducted has the objective to verify if the sequences identified are self-explainable or can be easily interpreted as activities, i.e., answering **RQ1**. To this end, the experiment showed that vAMoS was capable of identifying much longer motifs compared to LPM and was able to abstract a lot of behavior from low-level activities. The patterns identified by LPM are very short, therefore is challenging to recognize a meaningful behavior. Instead, the motif from vAMoS are clear in their structure, and can be intuitively matched with the performance of a specific activity. What is more, we observed that vAMoS was able to handle the variability. In fact, analyzing the traces in the event log in which every single motif was identified, we noticed that there was a strong similarity despite some discrepancies such as in the alternation of the order of two sensors.

Another key factor to focus on is the resulting event log, that is the abstract log. As indicated in **RQ2** we wanted to investigate if the algorithm was able to recognize behaviors and abstract them. The second evaluation conducted on the CASAS dataset shows the difference between the event log before and after the abstraction. The results of the abstraction are promising, since a high percentage of the event log is grouped into higher-level activities. Therefore, we can conclude that all research questions presented in Section 1 (i.e., **RQ1**, **RQ2**) can be positively answered.

Although the compelling results, the method proposed in this paper suffers from certain limitations. As introduced before, the procedures for calculating the set of candidate motifs as well as their verification are computationally intensive. During the application of the approach on a real dataset, the number of candidates (motifs for vAMoS, models for LPM) to be evaluated were remarkably high (i.e., millions of candidates). In this case, we made an assumption for the application of vAMoS, that is to keep the n-gram of the same length of the motif to reduce the number of candidates being verified each time. For this reason, it might be possible to introduce optimizations, yet these are beyond the scope of this work. For example, more optimized ways for constructing the set of candidates could be envisioned where only candidates with more likelihood of becoming verified are considered. It is important to highlight, however, that not only candidates perfectly seen in the log should be considered (though this can be done, by assigning the value of the n-gram length equal to the motif length), otherwise, there is the risk of missing some. Furthermore, the number of recognized motifs may be considerable, due to a non-optimized configuration of the parameters. In order for the algorithm to produce a limited number of motifs, which can be handled by an expert, some fine-tuning is required.

5 Conclusion and Future Work

In this paper, we presented a trace-based approach for event abstraction, named vAMoS. The fundamental idea of vAMoS, which leverages the qPMS algorithm, is based on the identification of recurring motifs that can be observed in a given percentage of the traces of the event log, up to a certain dissimilarity. The novelty introduced in the paper lies in the generalization of the qPMS algorithm (e.g., alphabet, candidates, distance, costs) as well as in the application scenario. The algorithm uses n-grams, as small observed sequences, that are combined for the construction of the set of candidate motifs. The motifs are evaluated based on a distance function that makes use of a cost relation. The algorithm is implemented as a Java application and it has been tested on a real dataset, in comparison with the state of the art.

In future work, we plan to investigate further improvements to the algorithm, in particular connected to the limitations introduced above. For example, having a more efficient identification of candidate motifs, where only viable candidates are considered. The labeling of the abstractions also represents an important future endeavor of our work.

References

1. Van der Aalst, W.M.: Process mining: data science in action. Springer (2016)
2. Baier, T., Di Ciccio, C., Mendling, J., Weske, M.: Matching events and activities by integrating behavioral aspects and label analysis. SoSyM **17**(2), 573–598 (2018)
3. Carmona, J., van Dongen, B., Solti, A., Weidlich, M.: Conformance checking. Springer (2018)

4. Cook, D.J.: Learning setting-generalized activity models for smart spaces. IEEE intelligent systems **2010**(99), 1 (2010)
5. Di Federico, G., Burattin, A., Montali, M.: Human behavior as a process model: Which language to use? In: IT-BPM. pp. 18–25. CEUR-WS (2021)
6. Janiesch, C., et al.: The Internet of Things Meets Business Process Management: A Manifesto. IEEE Systems, Man, and Cybernetics Magazine **6**(4), 34–44 (2020)
7. de Leoni, M., Dündar, S.: Event-log abstraction using batch session identification and clustering. In: Proceedings of the ACM SAC. pp. 36–44 (2020)
8. Leotta, F., Mecella, M., Sora, D.: Visual process maps: A visualization tool for discovering habits in smart homes. Journal of Ambient Intelligence and Humanized Computing **11**(5), 1997–2025 (2020)
9. Mannhardt, F., Leoni, M.d., Reijers, H.A., Van Der Aalst, W.M., Toussaint, P.J.: From low-level events to activities-a pattern-based approach. In: International conference on business process management. pp. 125–141. Springer (2016)
10. Mannhardt, F., Tax, N.: Unsupervised event abstraction using pattern abstraction and local process models (2017)
11. Melman, P., Roshan, U.W.: K-means-based feature learning for protein sequence classification. In: Proceedings of BICOB (2018)
12. Nicolae, M., Rajasekaran, S.: qpms9: an efficient algorithm for quorum planted motif search. Scientific reports **5**(1), 1–8 (2015)
13. Tax, N., Sidorova, N., Haakma, R., van der Aalst, W.M.: Mining local process models. Journal of Innovation in Digital Ecosystems **3**(2), 183–196 (2016)
14. Van Eck, M.L., Sidorova, N., Van der Aalst, W.M.: Enabling process mining on sensor data from smart products. In: Proceedings of RCIS. pp. 1–12. IEEE (2016)
15. van Zelst, S.J., Mannhardt, F., de Leoni, M., Koschmider, A.: Event abstraction in process mining: literature review and taxonomy. Granul. Comput. **6**(3) (2021)