Clustering analysis and frequent pattern mining for process profile analysis: an exploratory study for object-centric event logs

Elio Ribeiro Faria Junior[1,2][0000−0002−4358−5999], Thais Rodrigues Neubauer[1][0000−0003−4806−0830], Marcelo Fantinato[1][0000−0001−6261−1497], and Sarajane Marques Peres[1][0000−0003−3551−6480]

[1]Universidade de São Paulo, 03828-000 - São Paulo, SP, Brazil
[2]Universidade do Contestado, 89300-000 - Mafra, SC, Brazil
{elioribeirofaria, thais.neubauer, m.fantinato, sarajane}@usp.br

Abstract. Object-centric event log is a format for properly organizing information from different views of a business process into an event log. The novelty in such a format is the association of events with objects, which allows different notions of cases to be analyzed. The addition of new features has brought an increase in complexity. Clustering analysis can ease this complexity by enabling the analysis to be guided by process behaviour profiles. However, identifying which features describe the singularity of each profile is a challenge. In this paper, we present an exploratory study in which we mine frequent patterns on top of clustering analysis as a mechanism for profile characterization. In our study, clustering analysis is applied in a trace clustering fashion over a vector representation for a flattened event log extracted from an object-centric event log, using a unique case notion. Then, frequent patterns are discovered in the event sublogs associated with clusters and organized according to that original object-centric event log. The results obtained in preliminary experiments show association rules reveal more evident behaviours in certain profiles. Despite the process underlying each cluster may contain the same elements (activities and transitions), the behaviour trends show the relationships between such elements are supposed to be different. The observations depicted in our analysis make room to search for subtler knowledge about the business process under scrutiny.

Keywords: Object-Centric Event Log · Process Mining · Trace Clustering · Association Rules.

1 Introduction

Process mining aims to discover knowledge about how business processes actually occur [1]. This knowledge is primarily revealed by process model discovery and conformance checking techniques but can also come from modeling descriptive or predictive tasks. Once discovered, the knowledge is used for process improvement, through optimization of procedures in the organizations proposed either via human decisions or via automated prescriptive analysis.

For about 20 years, the main input for process mining was event logs derived from a single business process notion, herein called traditional event logs. For instance, in an ITIL framework context, one would only consider events related to activities in the "incident" life cycle, leaving out the life cycle of a "problem" to which the incident relates. Recently, the Process and Data Science Group from RWTH Aachen University [8] proposed a new event log format for recording events related to the life cycle of over one process notion. The new format is called object-centric event log (OCEL). The use of this format is expanding rapidly due to scientific community efforts to adapt process mining techniques to work with it [2, 4, 3]. One challenge brought by this format is how to overcome the increase in complexity it causes. Spaghetti-style process models [1] are even more often obtained from OCEL-type event logs.

One way used in process mining with traditional event logs to deal with process model complexity is to cluster process instances. Through clustering analysis [14, 6], the discovered process behaviour profiles provide knowledge about process particularities that simplifies subsequent applications of process mining techniques. For a proper profile analysis, the characterization of each profile is an important step that can be conducted by mining frequent patterns [10] existing in each profile or subset of profiles. In this paper, we describe an exploratory study consisted of applying clustering analysis followed by frequent pattern mining to facilitate the analysis of processes related to OCEL-type event logs. Even though the study was carried out on a synthetic and relatively simple event log, the results show the usefulness of the applied approach. The feasibility was also proved since the results brought knowledge for profile characterization in a semi-automated way – a business expert is required to extract semantic information from the frequently mined patterns. To the best of our knowledge, there is only one recent work [9] related to clustering analysis in OCEL-type event logs. In that work, the authors present a clustering strategy considering control-flow information and attributes values, while our approach focus on activities and transition occurrences. Besides, our approach goes beyond the discovery of clusters and presents a semi-automated way of characterizing them, while in [9] the authors present process models discovered upon clusters for visual analysis purposes. Both studies apply cluster analysis to flattened event logs, derived from different OCEL-type event logs, and present statistics that, although distinct, address the simplification provided by the resulting clusters.

This paper is organized as follows: Section 2 presents theoretical background on OCEL, clustering analysis and frequent pattern mining; Section 3 provides information on our exploratory study; Section 4 discusses the results related to cluster analysis, and the knowledge extracted from the mined frequent patterns; Section 5 resumes the contribution of our paper and highlights the research avenues raised from the exploratory study.

2 Theoretical Background

This section summarizes the theoretical concepts used in the exploratory study.

2.1 Object-Centric Event Logs

The process mining field aims to explore the knowledge latent to an event log generated from a business process execution. A traditional event log, as established by van der Aalst [1], contains data about events arising from the execution of activities of a specific *business case*. For example, an event log may concern the life cycle of purchase orders in an e-commerce system, while another event log concentrates data on the life cycle of deliveries of products purchased in this system. Therefore, each of these event logs assumes a *case notion*. However, the analysis provided by each of these event logs does not consider these life cycles are related, and a phenomenon observed in one life cycle may be because of facts occurred in the other life cycle. To overcome this limited and possibly incomplete analysis, the object-centric event logs were introduced [8]. In this new paradigm, multiple notions of cases are represented with information about the relationship between events and objects (e.g., orders, products, deliveries, etc.). According to van der Aalst [1] and van der Aalst and Berti [2], traditional event logs and object-centric event logs are defined as follows:

▷ *a traditional event log L is a set of cases, or process instances, $L \subseteq \mathcal{C}$, being $\mathcal{C}$ a universe of cases with respect to a unique business case notion. Cases may be characterized by descriptive attributes, among which one is mandatory - the trace. A* trace *corresponds to a finite sequence of events $\sigma \in \mathcal{E}^*$, being $\mathcal{E}^*$ a non-empty universe of events. An* event e *is the occurrence of a process activity at a given time. Events may be characterized by attributes such as timestamp, activity label, resource, cost, etc. An event appears at most once in L.*

▷ *a object-centric event log L_{oc} is a set of events $e_{oc} \in \mathcal{E}_{oc}$ partially ordered in time, such that $e_{oc} =$ (*ei, act, time, omap, vmap*), and* ei *is an event identifier,* act *is an activity name,* time *is a timestamp,* omap *is a mapping indicating which object is included for each type of object in L_{oc} and* vmap *is a mapping indicating the values assumed by each attribute in L_{oc}. Although a L_{oc} is partially ordered, for practical effects, a time-based total order is applied*[1].

The diversity of information in the object-centric event log increases the complexity of the associated analyses, prompting the search for strategies to simplify the event log without losing relevant information. In [2], the authors present a suitable way of filtering the object-centric event log. In the proposed strategy, the authors suggest filtering out specific "activity - object type" combinations. Following this strategy, chosen objects and activities related to them are suppressed from the log without harmful effect to activities and relationships referring to other types of objects. Consequently, the event log can be reduced in relation to the number of objects it contains, or events related to infrequent activities can be deleted. Simplification by "activity - object type" combinations filtering is a convenient alternative to flattening the log or to separately analyzing each type of object. However, selecting the "activity - object type" combination to be filtered requires *a priori* knowledge of what is relevant for the intended analysis.

[1] There are definitions that assume the total order ($\leq$) for L_{oc} [4, 9]. Such a definition states that L_{oc} is a tuple of events with total order.

2.2 Clustering Analysis

The task of clustering data is defined as a separation of data points into clusters according to a similarity metric. The goal is to allocate similar data points to the same cluster and dissimilar data points into different clusters. Although there are methods as density criterion or mutual information, distance measures based on the values of the features describing the data points are commonly used as similarity metrics [10]. The resolution of clustering tasks reveals descriptive information about the data set under analysis in an unsupervised form.

An assortment of clustering algorithms can be found in the literature. One category of fundamental clustering methods is the hierarchical methods, which partition the data into groups at different levels, as in a hierarchy. The provided hierarchical representation of the data points enables identifying that groups of a certain level can be further divided into respective subgroups. Hierarchical clustering methods are divided into agglomerative and divisive. We are interested in the first one, which is described as follows [10]:

▷ *the agglomerative clustering method starts at a level in which each data point forms a cluster and in each next level, the clusters are merged according to a similarity metric; by the end, it reaches a level in which there is only one cluster compound by all the data points. This method relies on measuring the distance being clusters to decide when to merge. The way of comparing the distance between clusters has to be defined, as a cluster is a set of objects. Possible ways are: single-linkage; complete-linkage; average-linkage; Ward's method.*

In process mining, we have observed applications of clustering analysis in the form of trace clustering [14, 6]. Trace clustering strategies can be divided into three non-excluding categories [11]: *trace sequence similarity*, *model similarity* and *feature vector similarity*. We are interested in the latter strategy:

▷ *trace clustering based on feature vector similarity relies on mapping of traces to a vector space by extracting features from a specific profile (such as activity, transition, performance or resource profile [6]). Clustering algorithms are applied on such vector representation to analyze similarities and group data points.*

2.3 Frequent pattern mining

Patterns such as itemsets, subsequences, substructures and association or correlation rules that frequently appear in a data set are called *frequent patterns*. Frequent pattern mining is a data mining task whose aim is to mine relationships in a given data set [10]. Mining frequent itemsets enables the discovery of associations and correlations among data. In this paper, we are interested in mining itemsets and association rules:

▷ *an itemset refers to a set of items. An itemset that contains* k *items is a* k*-itemset. When an itemset is frequent in a given data set, it can be called frequent itemset. If we have a frequent* 2*-itemset as* {milk, bread}*, it means that such itemset is frequent in the corresponding data set.*

▷ *an association rule defines an if-then association between itemsets organized in the antecedent and the consequent of such rule. The rule* milk ⇒ bread *means that if a customer buys milk then they also buy bread, frequently.*

To identify which of the mined patterns are useful, the support is defined as an interestingness measure. The support informs the percentage of all the existing transactions in which the pattern occurred. For association rules, on top of the support measure, the confidence measure is defined as an interestingness measure to bring how certain is the rule. For instance, for the association rule *milk* ⇒ *bread*: a support of 10% means that this rule occurs in 10% of the transactions (e.g., all the sold baskets); and a confidence of 60% means that in 60% of the baskets in which there is milk, there is also bread. Typically, domain experts[2] define a minimum support threshold and a minimum confidence threshold to filter the useful rules [10]. The classic algorithm *Apriori* [5] is widely used for mining frequent patterns. This algorithm is based on the item's anti-monotonicity property. In the first phase of this algorithm, such a property allows an efficient implementation for the frequent itemsets search. Frequent itemsets will compose association rules mined in its second phase.

3 Exploratory Study

Figure 1 depicts the sequence of procedures performed in the exploratory study, the resources applied (material and human resources) and the artifacts created during the study. This exploratory study comprises two phases: in the former, clustering analysis is used to discover existing behaviour profiles in the business process under scrutiny; in the latter, discovered profiles are explored through frequent pattern analysis, and the itemsets and association rules identified as useful and meaningful are used to provide knowledge about the profiles.

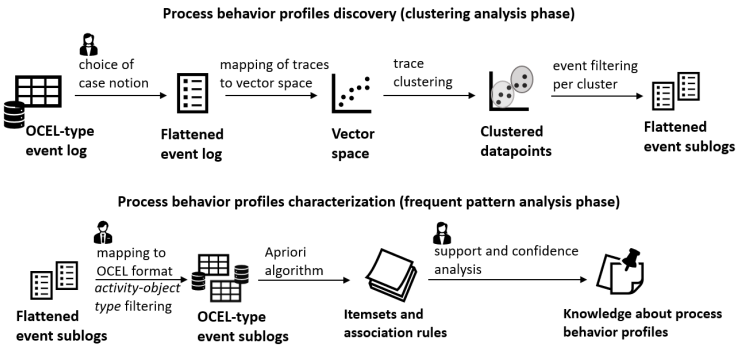
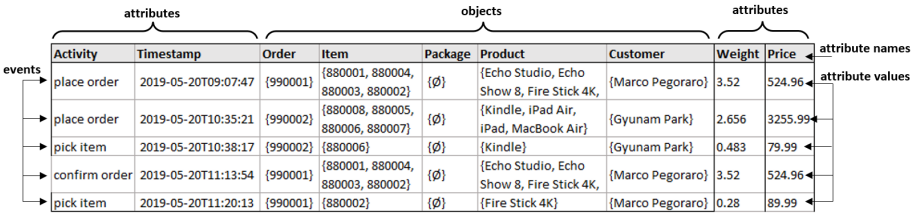


Fig. 1. Workflow followed in the exploratory study

[2]In this paper, the authors the authors played the role of domain experts.

3.1 Event log

The input to our study is a synthetic object-centric event log referring to an "order management" process [8, 9].[3,4] The process underlying the event log performs 11 activities on five types of objects (*orders*, *items*, *packages*, *customers*, and *products*). The execution registered in the event log comprises 22,367 events and 11,522 objects. Figure 2 represents an excerpt of this event log with all objects and attributes. We did not use the objects *product* and *customer*, and the attributes *price* and *weight*, since they do not represent an opportunity for control-flow perspective of analysis[5].



Activity	Timestamp	Order	Item	Package	Product	Customer	Weight	Price
place order	2019-05-20T09:07:47	{990001}	{880001, 880004, 880003, 880002}	{Ø}	{Echo Studio, Echo Show 8, Fire Stick 4K,	{Marco Pegoraro}	3.52	524.96
place order	2019-05-20T10:35:21	{990002}	{880008, 880005, 880006, 880007}	{Ø}	{Kindle, iPad Air, iPad, MacBook Air}	{Gyunam Park}	2.656	3255.99
pick item	2019-05-20T10:38:17	{990002}	{880006}	{Ø}	{Kindle}	{Gyunam Park}	0.483	79.99
confirm order	2019-05-20T11:13:54	{990001}	{880001, 880004, 880003, 880002}	{Ø}	{Echo Studio, Echo Show 8, Fire Stick 4K,	{Marco Pegoraro}	3.52	524.96
pick item	2019-05-20T11:20:13	{990001}	{880002}	{Ø}	{Fire Stick 4K}	{Marco Pegoraro}	0.28	89.99

Fig. 2. "Order management" object-centric event log excerpt

Figure 3 shows the process model discovered from the filtered "order management" event log, represented by a direct flow graph. Activities and transitions are colored according to the object they refer to: green refers to object *order*; pink refers to object *item*; red refers to object *package*. Although a visual analysis of the process behaviour is possible in this case, it can be tiring and imprecise, especially when more complex processes are analyzed, justifying the application of strategies to simplify the knowledge discovery on the process under scrutiny.

3.2 Process behaviour profiles discovery: clustering analysis phase

The first phase of our study comprises the following procedures: choice of case notion; mapping traces to vector space; trace clustering; and event filtering per cluster. All procedures are described in this section.

Choice of case notion: The profile discovery proposed relies on a trace-based clustering analysis. Thus, we need to define a case notion (a *business case notion*, cf. Section 2.1) for establishing traces and create a flattened event log. We applied the case notion referring to the object type *order*. Since this object type is the only one related to all events in the event log, choosing such an object as case notion allowed that profile discovery considered information of all events.

[3] We used the JSON-OCEL serialized representation of the event log.

[4] `http://ocel-standard.org/1.0/running-example.jsonocel.zip`

[5] Refer to [1] for information about control-flow perspective of analysis.

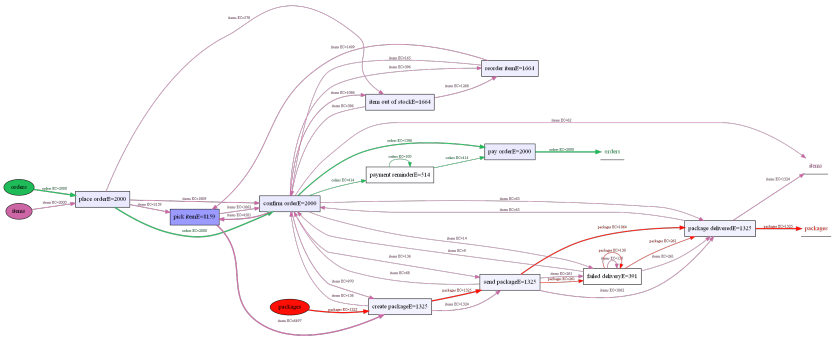


Fig. 3. Process model discovered from the filtered "order management" event log (process model discovered by using the package PM4Py for Python [7])

Mapping of traces to vector space: We represented traces in a vector space using two sets of descriptive features: the occurrence of activities in a trace (activity-based representation); the occurrence of transitions in a trace (transition-based representation). The former does not consider the order in which activities occur, but provides a representation that incorporates similarity in the resulting data points (e.g., traces with the same activities but not the same execution order are mapped to the same data point). The latter represents the partial order in which activities occur, emphasizing a process-aware similarity analysis.

Trace clustering: Trace clustering was applied using an agglomerative hierarchical clustering algorithm [13][6], with Ward as the linkage method, Euclidean distance as similarity metric and number of clusters set to six. The authors' experience in trace clustering showed the Ward's method allows finding clusters with slightly higher quality than using other linkage methods. The Euclidean distance was chosen as the first option for exploration in this study. We tested the number of clusters ranging from three to six. A profile associated with the "value chain" of the business process under scrutiny was found with five and six clusters considering the activity-occurrence representation; the number six was chosen to maximize the number of profiles for analysis. The same number was used with transition-occurrence representation for the sake of uniformity.

Event filtering per cluster: Once the trace clusters are built, we separate the events associated with each cluster into independent files, the flattened sublogs.

3.3 Process behaviour profiles characterization: frequent pattern analysis phase

The second phase of our study comprises the following procedures: mapping to OCEL format and *activity-object type* filtering; Apriori algorithm application; and support and confidence analysis. All procedures are described in this section.

[6] sklearn.cluster.AgglomerativeClustering: `http://https://scikit-learn.org/`

Mapping to OCEL format and activity-object type filtering: Flattened sublogs must be mapped back into OCEL-type event sublogs considering both the notion of case previously chosen and the *activity-object type* filter that relates activities to object types appropriately as suggested in [2]. The selection of *activity-object type* combinations to be used requires a business process-oriented decision making, usually carried out by a business expert.

Apriori algorithm: For discovery of frequent patterns, the classic Apriori algorithm[7] was applied on each OCEL-type event sublog, considering the activities and transitions associated with each object type (*order*, *item*, *package*) separately. 18 sets of itemsets and association rules were created (i.e. one set per cluster per object). The input to the algorithm is a matrix of occurrences of activities (or transitions) in the object-type life cycle. The Apriori algorithm runs were performed with *minimum support* = 0.05 (for both itemsets and association rules) and *minimum confidence* = 0.9 (such values were set by experimentation).

Support and confidence analysis: The frequent patterns for each of the six clusters were compared following a one-versus-all strategy. This strategy enables selecting patterns which differs one cluster from the other clusters. Then, the selected frequent patterns were (manually) analyzed to extract expert knowledge about the discovered process behaviour profiles.

4 Analysis of results

The first phase of our study aimed to reveal process behaviour profiles that provide simpler contexts for analysis and knowledge discovery than the context provided by the full event log. Table 2 and Table 3 show descriptive statistics for supporting analysis about simplicity of the context referring to each discovered profile (i.e. each cluster), considering activity-based and transition-based representation for traces. The descriptive statistics for the full event log were presented in [2] and are reproduced here for comparison purposes (see Table 1). Statistics refers to the *average* and *maximum number* of objects per event[8]. In these tables, "O", "I" and "P" stand for *orders*, *items* and *packages* respectively.

The comparison of statistics shows clustering generates more simplified contexts in two aspects: some clusters represent process profiles in which certain objects do not appear related to events of certain activities (e.g., there are no items associated with the activity "item out of stock" in the process profile of the clusters *a1*, *a4* and *a5*, showing these profiles do not suffer from the problem of an item not being found in stock while an order is processed); the occurrence of a maximum number of objects related to events of certain activity is lower in certain process profiles (e.g., fewer items enter the orders allocated in cluster *a1* and *t4* - citing only two clusters). However, in general, the averages of objects per event increase, as the number of events present in the clusters decreases.

[7]Package Mlxtend: `https://rasbt.github.io/mlxtend/`.

[8]The statistic *minimum number* was suppressed from the tables 1, 2 and 3 for simplicity. *Minimum* number = 1 if *maximum* number ≥ 1, and = 0 otherwise.

Table 1. Descriptive statistics about the full event log [2].

Activities	O	I	P	Activities	O	I	P
place order	1.0, 1	4.0 15	0.0 0				
confirm order	1.0, 1	4.0 15	0.0 0	pay order	1.0 1	4.0 15	0.0 0
item out of stock	1.0, 1	1.0 1	0.0 0	create package	3.2 9	6.2 22	1.0 1
reorder item	1.0, 1	1.0 1	0.0 0	send package	3.2 9	6.2 22	1.0 1
pick item	1.0, 1	1.0 1	0.0 0	failed delivery	3.2 8	6.0 18	1.0 1
payment reminder	1.0, 1	4.2 14	0.0 0	package delivered	3.2 9	6.2 22	1.0 1

In the second phase, we mined and analyzed the frequent patterns to reveal knowledge about the process profiles, alleviating the need to discover and inspect process models related to each sublog. We organized the analyses considering the two matrices of occurrences used as input for the Alpha algorithm.

Matrix of activity occurrences: We identified 13 association rules not common to all clusters. All rules involved 1-itemsets, achieved maximum confidence and the itemsets allocated to their consequents have maximum support. Thus, the rules analysis was reduced to the analysis of the support of itemsets allocated to their antecedents. The relevant knowledge that characterizes the profiles are:

- payment reminders occur on all process instances in the profiles *a0* and *a1*;
- delivery failures occur in part of the process instances in profiles *a0*, *a1*, *a2* and *a5*, with emphasis on profile *a5* in which $\approx 60\%$ of the process instances present the occurrence of such a problem;
- out-of-stock items are observed in $\approx 30\%$ of process instances in profiles *a0*, *a2* and *a3*.

The discovered frequent patterns concern the occurrence of activities that indicate some kind of problem in the order history. None of such patterns were highlighted for the profile *a4*. All association rules highlighted to profile *a4* achieve maximum support and maximum confidence and do not involve activities related to failures or out-of-stock items. In view of these findings, we deduced the profile *a4* concerns the process instances that follow the process's "value chain", or follow behaviours very close to it. To validate the deduction, we discovered the process model associated with this profile (Figure 4).

Matrix of transition occurrences: We identified 26 association rules not common to all clusters. All rules involved 1-itemsets, 17 rules achieved the maximum confidence, the minimum confidence achieved was 0.91, and in two rules the consequent is not composed by an itemset with maximum support. The relevant knowledge that characterize the profiles are:

- payment of orders without sending a reminder is a majority behaviour (occurs in $\approx 80\%$ to $\approx 98\%$ of process instances) in five profiles (*t0*, *t1*, *t2*, *t4* and *t5*);

Table 2. Descriptive statistics about profiles discovered upon activity-based representation for traces. Statistics showing simplification are in bold.

	O	**I**	**P**	**O**	**I**	**P**	**O**	**I**	**P**
Activities	**Cluster a0**			**Cluster a1**			**Cluster a2**		
place order	1.0, 1	4.7, **14**	0.0, 0	1.0, 1	**3.3, 7**	0.0, 0	1.0, 1	4.9, 15	0.0, 0
confirm order	1.0, 1	4.7, **14**	0.0, 0	1.0, 1	**3.3, 7**	0.0, 0	1.0, 1	4.9, 15	0.0, 0
item out of stock	1.0, 1	1.0, 1	0.0, 0	**0.0, 0**	**0.0, 0**	0.0, 0	1.0, 1	1.0, 1	0.0, 0
reorder item	1.0, 1	1.0, 1	0.0, 0	**0.0, 0**	**0.0, 0**	0.0, 0	1.0, 1	1.0, 1	0.0, 0
pick item	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0
payment reminder	1.0, 1	4.8, 14	0.0, 0	1.0, 1	3.3, **7**	0.0, 0	**0.0, 0**	**0.0, 0**	0.0, 0
pay order	1.0, 1	4.7, **14**	0.0, 0	1.0, 1	3.3, **7**	0.0, 0	1.0, 1	4.9, 15	0.0, 0
create package	3.9, 9	7.0, 22	1.0, 1	3.9, 9	7.3, **20**	1.0, 1	3.7, 9	6.6, 22	1.0, 1
send package	3.9, 9	7.0, 22	1.0, 1	3.9, 9	7.3, **20**	1.0, 1	3.7, 9	6.6, 22	1.0, 1
failed delivery	3.8, 8	6.7, 18	1.0, 1	3.8, 8	7.2, 18	1.0, 1	3.2, 8	6.1, 18	1.0, 1
package delivered	3.9, 9	7.0, 22	1.0, 1	3.9, 9	7.3, **20**	1.0, 1	3.7, 9	6.6, 22	1.0, 1
	Cluster a3			**Cluster a4**			**Cluster a5**		
place order	1.0, 1	4.4, **14**	0.0, 0	1.0, 1	3.2, 9	0.0, 0	1.0, 1	**3.5, 10**	0.0, 0
confirm order	1.0, 1	4.4, **14**	0.0, 0	1.0, 1	3.2, 9	0.0, 0	1.0, 1	**3.5, 10**	0.0, 0
item out of stock	1.0, 1	1.0, 1	0.0, 0	**0.0, 0**	**0.0, 0**	0.0, 0	**0.0, 0**	**0.0, 0**	0.0, 0
reorder item	1.0, 1	1.0, 1	0.0, 0	**0.0, 0**	**0.0, 0**	0.0, 0	**0.0, 0**	**0.0, 0**	0.0, 0
pick item	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0
payment reminder	**0.0, 0**	**0.0, 0**	0.0, 0	**0.0, 0**	**0.0, 0**	0.0, 0	**0.0, 0**	**0.0, 0**	0.0, 0
pay order	1.0, 1	4.4, 14	0.0, 0	1.0, 1	**3.2, 9**	0.0, 0	1.0, 1	**3.5, 10**	0.0, 0
create package	3.6, 9	6.6, 22	1.0, 1	3.8, 9	7.2, 22	1.0, 1	3.9, 9	7.2, **21**	1.0, 1
send package	3.6, 9	6.6, 22	1.0, 1	3.8, 9	7.2, 22	1.0, 1	3.9, 9	7.2, **21**	1.0, 1
failed delivery	**0.0, 0**	**0.0, 0**	0.0, 0	**0.0, 0**	**0.0, 0**	0.0, 0	3.8, 8	7.1, 18	1.0, 1
package delivered	3.6, 9	6.6, 22	1.0, 1	3.8, 9	7.2, 22	1.0, 1	3.9, 9	7.2, **21**	1.0, 1

- reminders before the payment of an order is made occur in ≈ 99% of process instances allocated to the profile *t3*;
- repeated payment reminders occur only in profile *t3* and represent ≈ 21% of the processes instances allocated in this profile;
- in the profiles *t0*, *t2*, *t3* and *t5*, there are orders (≈ 30%, 15%, 6% and 8% respectively) in which the observation related to out-of-stock items occurs after the order is confirmed;
- although not really significant (rule with support from ≈ 0.0 to ≈ 13%), delivery failures are pointed at least twice in process instances of four profiles (*t1*, *t2*, *t4* and *t5*);
- packages successfully delivered on the first attempt occur in process instances allocated in all profiles (in 71/74/76/80/83/88% of process instances allocated respectively to profiles *t1*, *t2*, *t5*, *t4*, *t0*, and *t3*).

5 Final remarks

In this paper, we introduce an approach to simplify the context of analysis related to OCEL-type event logs and present an exploratory experiment performed on

Table 3. Descriptive statistics about profiles discovered upon transition-based representation for traces. Statistics showing simplification are in bold.

	O	**I**	**P**	**O**	**I**	**P**	**O**	**I**	**P**
Activities	**Cluster t0**			**Cluster t1**			**Cluster t2**		
place order	1.0, 1	4.0, **14**	0.0, 0	1.0, 1	**3.7, 11**	0.0, 0	1.0, 1	5.1, 15	0.0, 0
confirm order	1.0, 1	4.0, **14**	0.0, 0	1.0, 1	**3.7, 11**	0.0, 0	1.0, 1	5.1, 15	0.0, 0
item out of stock	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0
reorder item	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0
pick item	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0
payment reminder	1.0, 1	4.2, 14	0.0, 0	1.0, 1	**3.7, 10**	0.0, 0	1.0, 1	5.1, **13**	0.0, 0
pay order	1.0, 1	4.0, 14	0.0, 0	1.0, 1	**3.7, 11**	0.0, 0	1.0, 1	5.1, **15**	0.0, 0
create package	3.8, 9	6.8, 22	1.0, 1	3.7, 9	7.0, 22	1.0, 1	3.6, 9	6.6, 22	1.0, 1
send package	3.8, 9	6.8, 22	1.0, 1	3.7, 9	7.0, 22	1.0, 1	3.6, 9	6.6, 22	1.0, 1
failed delivery	3.7, **7**	**5.6, 17**	1.0, 1	3.7, 8	6.8, **17**	1.0, 1	3.4, 8	6.3, 18	1.0, 1
package delivered	3.8, 9	6.8, 22	1.0, 1	3.7, 9	7.0, 22	1.0, 1	3.6, 9	6.6, 22	1.0, 1
	Cluster t3			**Cluster t4**			**Cluster t5**		
place order	1.0, 1	**3.7, 11**	0.0, 0	1.0, 1	**3.1, 10**	0.0, 0	1.0, 1	**3.9, 13**	0.0, 0
confirm order	1.0, 1	**3.7, 11**	0.0, 0	1.0, 1	**3.1, 10**	0.0, 0	1.0, 1	**3.9, 13**	0.0, 0
item out of stock	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0
reorder item	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0
pick item	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0	1.0, 1	1.0, 1	0.0, 0
payment reminder	1.0, 1	**3.8, 11**	0.0, 0	1.0, 1	**3.5, 5**	0.0, 0	1.0, 1	6.0, **6**	0.0, 0
pay order	1.0, 1	**3.7, 11**	0.0, 0	1.0, 1	**3.1, 10**	0.0, 0	1.0, 1	3.9, **13**	0.0, 0
create package	3.8, **8**	6.9, **20**	1.0, 1	4.0, 9	7.3, 22	1.0, 1	4.3, **8**	7.7, **21**	1.0, 1
send package	3.8, **8**	6.9, **20**	1.0, 1	4.0, 9	7.3, 22	1.0, 1	4.3, **8**	7.7, **21**	1.0, 1
failed delivery	3.5, **6**	6.3, **13**	1.0, 1	3.6, 8	6.8, 18	1.0, 1	4.2, **7**	7.7, **16**	1.0, 1
package delivered	3.8, **8**	6.9, **20**	1.0, 1	4.0, 9	7.3, 22	1.0, 1	4.3, **8**	7.7, **21**	1.0, 1

a synthetic event log. The preliminary results show the usefulness and feasibility of our approach. The approach is useful because it allows extracting knowledge capable of highlighting, in each profile, characteristics that can direct subsequent in-depth analyses. It is feasible because, even in a low-complexity event log with little potential for profiling, it was possible to find and characterize a set of profiles. However, this is an exploratory study limited mainly by the arbitrary choice of some parameters, such as the business case notion, the similarity metric or the number of clusters. In addition, the experiment considered a single event log, which undermines both statistical and analytical generalizations. The execution of this study opened up research opportunities: extension of the frequent pattern mining to discover association rules that characterize profiles considering the relationship among the life cycles of different objects types; using of attributes referring to the business context and available in the OCEL-type event logs to enrich the relationships explored in the frequent pattern mining; adding frequent pattern mining outputs in tools for trace clustering visualization [12].

Acknowledgment. This study was partially supported by CAPES (Finance Code 001) and FAPESP (Process Number 2020/05248-4).

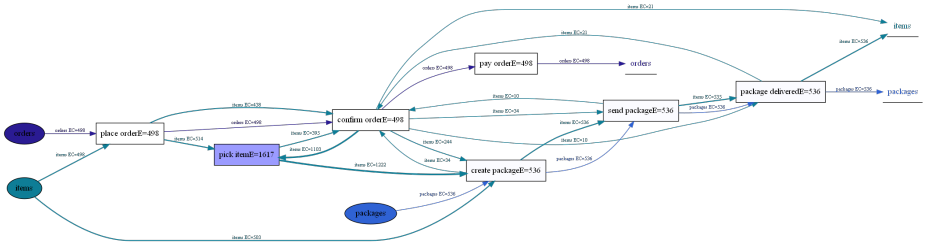


Fig. 4. Process model related to profile *a4* involving the "value chain": place order, pick item, confirm order, pay order, create package, send package, package delivered.

References

1. van der Aalst, W.M.P.: Process Mining - Data Science in Action. Springer (2016)
2. van der Aalst, W.M.P., Berti, A.: Discovering object-centric petri nets. Fundamenta Informaticae **175**(1–4), 1–40 (2020)
3. van der Aalst, W.M.P.: Object-centric process mining: Dealing with divergence and convergence in event data. In: Ölveczky, P.C., Salaün, G. (eds.) Software Engineering and Formal Methods. pp. 3–25. Springer (2019)
4. Adams, J.N., Van Der Aalst, W.M.: Precision and fitness in object-centric process mining. In: Proc. 3rd Int. Conf. on Process Mining. pp. 128–135 (2021)
5. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proc. 20th Int. Conf. on Very Large Data Bases. pp. 487–499 (1994)
6. Appice, A., Malerba, D.: A co-training strategy for multiple view clustering in process mining. IEEE Trans. Serv. Comput. **9**, 832–845 (2016)
7. Berti, A., van Zelst, S.J., van der Aalst, W.M.: PM4Py web services: Easy development, integration and deployment of process mining features in any application stack. In: Inf. Conf. on Business Process Managment (PhD/Demos) (2019)
8. Ghahfarokhi, A.F., Park, G., Berti, A., van der Aalst, W.M.P.: Ocel standard. `http://ocel-standard.org/` (2020)
9. Ghahfarokhi, A.F., Akoochekian, F., Zandkarimi, F., van der Aalst, W.M.P.: Clustering object-centric event logs (2022), `https://arxiv.org/abs/2207.12764`
10. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edn. (2011)
11. Lu, X.: Using behavioral context in process mining: exploration, preprocessing and analysis of event data. PhD thesis, Eindhoven University of Technology (2018)
12. Neubauer, T.R., Sobrinho, G.P., Fantinato, M., Peres, S.M.: Visualization for enabling human-in-the-loop in trace clustering-based process mining tasks. In: 5th IEEE Workshop on Human-in-the-Loop Methods and Future of Work in BigData. pp. 3548–3556 (2021)
13. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. J. of Machine Learning Research **12**, 2825–2830 (2011)
14. Song, M., Gunther, C.W., van der Aalst, W.M.P.: Trace clustering in process mining. In: BPM Workshops. pp. 109–120. Springer, Berlin, Germany (2008)