Clustering Object-Centric Event Logs

Anahita Farhang Ghahfarokhi[1,*], Fatemeh Akoochekian[1], Fareed Zandkarimi[2] and Wil M. P. van der Aalst[1]

[1] *Process and Data Science, RWTH Aachen University, Aachen, Germany*

[2] *Chair of Enterprise Systems, University of Mannheim, Mannheim, Germany*

Keywords: Clustering, Object-Centric Process Mining, Convergence.

Abstract: Process mining provides various algorithms to analyze process executions based on event data. Process discovery, the most prominent category of process mining techniques, aims to discover process models from event logs. However, it leads to spaghetti models when working with real-life data. To reduce the complexity of process models, several clustering techniques have been proposed on top of event logs with a single case notion. However, in real-life processes often multiple objects are involved in a process. Recently, Object-Centric Event Logs (OCELs) have been introduced to capture the information of such processes, and several process discovery techniques have been developed on top of OCELs. Yet, the output of the discovery techniques leads to complex models. In this paper, we propose a clustering-based approach to cluster similar objects in OCELs to simplify the obtained process models. Using a case study of a real Business-to-Business (B2B) process, we demonstrate that our approach reduces the complexity of the models and generates coherent subsets of objects which help the end-users gain insights into the process.

1 INTRODUCTION

Process mining is a field of science bridging the gap between data-oriented analysis and process-oriented analysis, which aims to extract knowledge from event logs Van der Aalst (2016). Over time, different event log formats such as XES have been established to implement process mining techniques. The proposed standards assume a single case notion in the process. However, in real processes multiple objects interact with each other Berti et al. (2022); Esser and Fahland (2021); Ghahfarokhi and van der Aalst (2021), for example, considering a Purchase-to-Pay (P2P) process where *orders*, *items*, and *customers* are involved Ghahfarokhi et al. (2021a,b). Several process discovery techniques have been developed on top of event logs with multiple case notions Berti (2022); Berti and van der Aalst (2018); Cohn and Hull (2009); Lu et al. (2015); Meroni et al. (2018). For example, Object-Centric DFGs (OC-DFGs), used throughout this paper, are one of the object-centric process models developed on top of Object-Centric Event Logs (OCELs). An OC-DFG is a Directly-Follows Graph (DFG) where relations are colored based on object types Berti and van der Aalst (2018). Several examples of OC-DFGs are shown in the remainder. OC-DFGs are usually complex to interpret. Some research has been done to cluster OCELs and simplify OC-DFGs. In Faria Junior et al. (2023) authors used frequent pattern mining to enrich the OCEL and applied trace clustering on the flattened event log. In Jalali (2022), the author used Markov Directly-Follow Multigraph to cluster similar case notions. Additionally, a threshold-tuning algorithm was applied to identify distinct clusters that can be uncovered at various similarity levels. In this paper, we present a clustering-based approach, shown in Figure 1, which uses the relations between objects and events. First, we extracted an OCEL from a Business-to-Business (B2B) process. Then, we enriched the extracted OCEL with graph-related attributes. Afterward, we selected a clustering object type and applied data clustering algorithms to group similar objects. The challenge occurs when assigning events to the clusters. Here we propose two approaches:

- *Existence*: If we directly assign events to the clusters by considering that the event should contain at least one of the objects in the cluster, then the same event may appear in several clusters. For example, consider the B2B OCEL shown in Ta-

Ghahfarokhi, A., Akoochekian, F., Zandkarimi, F. and van der Aalst, W.
Clustering Object-Centric Event Logs.
DOI: 10.5220/0012123900003541
In *Proceedings of the 12th International Conference on Data Science, Technology and Applications (DATA 2023)*, pages 444-451
ISBN: 978-989-758-664-4; ISSN: 2184-285X

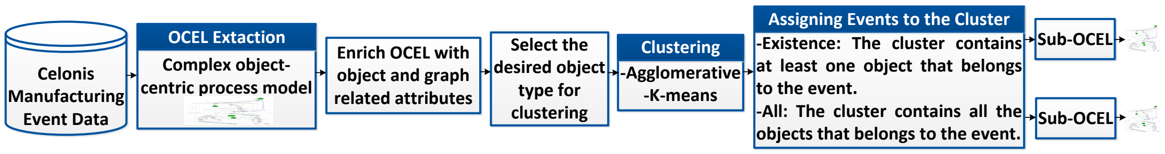

Celonis Manufacturing Event Data
OCEL Extaction
Complex object-centric process model
Enrich OCEL with object and graph related attributes
Select the desired object type for clustering
Clustering
-Agglomerative
-K-means
Assigning Events to the Cluster
-Existence: The cluster contains at least one object that belongs to the event.
-All: The cluster contains all the objects that belongs to the event.
Sub-OCEL
Sub-OCEL

Figure 1: Overview of the proposed framework.

Table 1: Informal representation of the events. Each row shows an event with its properties.

id	activity	timestamp	batch	order	customer	net price	gross price
e_1	*order creation*	*2020-04-13 11:20:01.527+01:00*	$\{\}$	$\{o_1\}$	$\{c_1\}$	*146.8*	*154.8*
e_2	*print of production order*	*2020-04-15 08:21:01.527+01:00*	$\{b_1,b_2\}$	$\{o_1\}$	$\{c_1\}$	*285.8*	*301.3*
e_3	*Loading*	*2020-05-09 08:22:01.527+01:00*	$\{b_1,b_3\}$	$\{o_1\}$	$\{c_1\}$	*272.47*	*312.4*
...	...	...	...	...	...	...	...

Table 2: Informal representation of the objects. Each row shows the properties of the objects.

id	type	treatment	workplace
b_1	*batch*	*painting*	*plant 1*
b_2	*batch*	*polishing*	*plant 1*
o_1	*order*		
...	...	...	...

bles 1 and 2, where *customer*, *order*, and *batch* are the possible case notions. When we apply clustering based on *batch*, then two *batches* in the same event may end up in two different clusters and be duplicated. For example, if b_1 and b_3 are in separate clusters, then e_3 is in both clusters. This is due to the convergence in OCELs where an event can contain multiple objects of the same type Ghahfarokhi et al. (2021b).

- *All*: We assign an event to a cluster if the cluster contains all the objects that exist in that event. In this approach, we may miss some events. As an example shown in Table 1, if b_1 and b_2 end up in different clusters, then we miss e_2, because all the batches of e_2 are not in the same cluster.

To evaluate the quality of the discovered OC-DFGs, we provide initial complexity measures for OC-DFGs. Using the proposed clustering techniques and quality measures, we achieved a set of meaningful OC-DFGs with almost the same fitness but less complexity in comparison with the initial model.

The remaining part of the paper is organized as follows. Section 2 presents the running example that is used throughout the paper. Then, in Section 3, we present some preliminary concepts. In Section 4, we discuss object profile extraction and enrichment. Afterward, in Section 5, we describe our proposed clustering-based approach in OCELs. Then, in Section 6, we provide some experiments on the running example using our approach in PM4PY. Finally, Section 7 concludes the paper and provides future work.

2 RUNNING EXAMPLE

To evaluate our approach, we have extracted OCEL from a real B2B process, anonymized and stored in Celonic Process Analytics Workspace. The respective industry performs surface treatment services such as coating and polishing for the automotive industry. Figure 2 presents the generic process routine. The process starts with the *order creation* activity. *Customers* send their *order* to the company and request for specific treatments. The orders will be split into *batches* to fit production machines. After applying the treatments, the respective *batches* of each *order* are packed together to be shipped back to the *customers*. Tables 1 and 2 show the extracted OCEL. The OC-DFG extracted from the whole process is an unreadable spaghetti model that does not give insights about the process. To derive simpler models, we cluster the OCEL into sub-logs. To apply clustering methods on objects in the OCEL, we extract object profiles from OCEL which we describe in the next section.

3 PRELIMINARIES

3.1 Object-Centric Event Logs

Definition 1 (Universes)**.** *We define the universes:*

- U_e *is the universe of event identifiers, e.g.,* $\{e_1, e_2, e_3\} \subseteq U_e$
- U_{act} *is the universe of activities, e.g.,* $\{order\ creation, last\ delivery\} \subseteq U_{act}$
- U_{att} *is the universe of attribute names, e.g.,* $\{gross\ price, net\ price\} \subseteq U_{att}$
- U_{val} *is the universe of attribute values, e.g.,* $\{200.0, 302.0, painting\} \subseteq U_{val}$
- U_{typ} *is the universe of attribute types., e.g.,* $\{string, integer, float\} \subseteq U_{typ}$

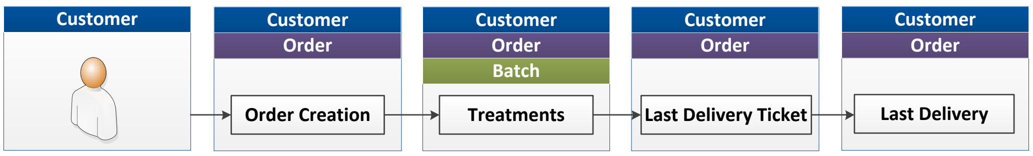


Figure 2: An abstract view of the extracted B2B process.

- U_o is the universe of object identifiers, e.g., $\{o_1, b_1\} \subseteq U_o$
- U_{ot} is the universe of objects types, e.g., $\{order, batch\} \subseteq U_{ot}$
- U_{timest} is the universe of timestamps, e.g., 2020-04-09T08:21:01.527+01:00 $\in U_{timest}$

Using the universes above, we define OCELs.

Definition 2 (Object-Centric Event Log). *An object-centric event log is a tuple* $L=(E, AN, AV, AT, OT, O, \pi_{typ}, \pi_{act}, \pi_{time}, \pi_{vmap}, \pi_{omap}, \pi_{otyp}, \pi_{ovmap}, \leq)$ *such that:*

- $E \subseteq U_e$ *is the set of event identifiers.*
- $AN \subseteq U_{att}$ *is the set of attributes names.*
- $AV \subseteq U_{val}$ *is the set of attribute values.*
- $AT \subseteq U_{typ}$ *is the set of attribute types. For example, the type of the attribute workplace in Table 2 is string.*
- $OT \subseteq U_{ot}$ *is the set of object types. For example, in Table 2, for the first object, the type is batch.*
- $O \subseteq U_o$ *is the set of object identifiers.*
- $\pi_{typ} : AN \cup AV \to AT$ *is the function associating an attribute name or value to its corresponding type. For example, in Table 1,* $\pi_{typ}(net\ price) = float$.
- $\pi_{act} : E \to U_{act}$ *is the function associating an event to its activity, e.g.,* $\pi_{act}(e_1) = order\ creation$.
- $\pi_{time} : E \to U_{timest}$ *is the function associating an event to a timestamp, e.g.,* $\pi_{time}(e_1)$ =2020-04-13 11:20:01.527+01:00 *in Table 1.*
- $\pi_{vmap} : E \to (AN \not\to AV)$ *is the function associating an event to its attribute value assignments, e.g.,* $\pi_{vmap}(e_1)(net\ price)$ =146.8 *in Table 1,*
- $\pi_{omap} : E \to \mathcal{P}(O)$ *is the function associating an event to a set of related object identifiers, e.g.,* $\pi_{omap}(e_1) = \{o_1, c_1\}$ *in Table 1.*
- $\pi_{otyp} \in O \to OT$ *assigns precisely one object type to each object identifier, e.g.,* $\pi_{otyp}(o_1) = order$.
- $\pi_{ovmap} : O \to (AN \not\to AV)$ *is the function associating an object to its attribute value assignments, e.g.,* $\pi_{ovmap}(b_1)(workplace) = plant\ 1$ *in Table 2.*
- $\leq$ *is a total order (i.e., it respects the anti-symmetry, transitivity, and connexity properties).*

Next, by selecting an object type that we aim to cluster, we transform an OCEL into a flattened log.

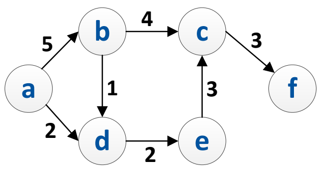


Figure 3: A directed graph.

Definition 3 (Ot-Flattened Log). *Let* $L = (E, AN, AV, AT, OT, O, \pi_{typ}, \pi_{act}, \pi_{time}, \pi_{vmap}, \pi_{omap}, \pi_{otyp}, \pi_{ovmap}, \leq)$ *be an OCEL, and* $ot \in OT$ *be an object type. We define ot-flattened log as* $FL(L, ot) = (E^{ot}, \pi^{ot}_{act}, \pi^{ot}_{time}, \pi^{ot}_{case}, \leq^{ot})$ *where:*

- $E^{ot} = \{e \in E \mid \exists_{o \in \pi_{omap}(e)}\ \pi_{otyp}(o) = ot\}$,
- $\pi^{ot}_{act} = \pi_{act|E^{ot}}$, *i.e.,* π_{act} *with the domain* E^{ot},
- $\pi^{ot}_{time} = \pi_{time|E^{ot}}$, *i.e.,* π_{time} *with the domain* E^{ot},
- *For* $e \in E^{ot}$, $\pi^{ot}_{case}(e) = \{o \in \pi_{omap}(e) \mid \pi_{otyp}(o) = ot\}$, *and*
- $\leq^{ot} = \{(e_1, e_2) \in \leq \mid \exists_{o \in O}\ \pi_{otyp}(o) = ot \land o \in \pi_{omap}(e_1) \cap \pi_{omap}(e_2)\}$

Using the flattened log, we extract object profiles that will be described in Section 4. To increase the number of object features, we use some graph-related attributes. Next, we describe the graph theory concepts that we used to enrich the OCELs.

Definition 4 (Directed Graph). *A directed graph is a pair* $G = (V, E)$ *where Bender and Williamson (2010):*

- V *is a set of vertices (nodes).*
- $E \subseteq \{(v_1, v_2) \in V \times V \mid v_1 \neq v_2\}$ *is a set of edges, which are ordered pairs of distinct vertices. In a weighted directed graph each node is assigned to a weight through the function* $f : E \to \mathbb{R}$.

Figure 3 shows and example of a weighted graph.

We have used graph features related to centrality (e.g., *in-degree centrality*, *out-degree centrality*, *closeness centrality*, and *harmonic centrality*) to enrich the object profiles. In graph theory, centrality is a number or ranking assigned to all nodes in a graph, indicating each node's position in the graph.

Definition 5 (In-Degree Centrality). *Let* $G = (V, E)$ *be a directed graph and* $v_1 \in V$. *Then we define* $deg_{in}(v_1)$ *as the number of incoming edges to* v_1, *i.e.,* $deg_{in}(v_1) = |\{(v, v') \in E \mid v' = v_1\}|$.

Example: In Figure 3, $deg_{in}(b) = 1$ *and* $deg_{in}(c) = 2$.

The formal definitions of other centrality features are explained in Ghahfarokhi et al. (2022). In the next section, we describe how we comprise object profiles and enrich them.

4 OBJECT PROFILES

In our clustering framework, the points to be clustered are object profiles. To start clustering, we preprocess the data and enrich it with additional features. First, we extract the object trace using the flattened log.

Definition 6 (Trace)**.** *Given an ot-flattened log* $FL = (E^{ot}, \pi^{ot}_{act}, \pi^{ot}_{time}, \pi^{ot}_{case}, \leq^{ot})$*, we define the following operations:*

- $\pi^{ot}_{act}(FL) = \{\pi^{ot}_{act}(e) \mid e \in E^{ot}\}$
- $\pi^{ot}_{case}(FL) = \cup_{e \in E^{ot}} \pi^{ot}_{case}(e)$
- *For* $c \in \pi^{ot}_{case}(FL)$, $case^{ot}_{FL}(c) = \langle e_1, \ldots, e_n \rangle$ *where:*
  - $\{e_1, \ldots, e_n\} = \{e \in E^{ot} \mid c \in \pi^{ot}_{case}(e)\}$
  - $\forall_{1 \leq i < n}\, e_i < e_{i+1}$
- *Given* $c \in \pi^{ot}_{case}(FL)$ *and* $case^{ot}_{FL}(c) = \langle e_1, \ldots, e_n \rangle$*,we define* $trace_{FL}(c) = \langle \pi^{ot}_{act}(e_1), \ldots, \pi^{ot}_{act}(e_n) \rangle$

To provide derived attributes, we create a graph based on the sequence of activities of each object.

Definition 7 (Trace Graph)**.** *Let* $FL = (E^{ot}, \pi^{ot}_{act}, \pi^{ot}_{time}, \pi^{ot}_{case}, leq^{ot})$ *be a flattened log and* $c \in \pi^{ot}_{case}(FL)$ *be an object. For the object trace* $trace_{FL}(c) = \langle a_1, \ldots, a_n \rangle$*, we define the corresponding directed weighted graph as:*

$G_{trace_{FL}(c)} = (V, E)$ *with the weight function* $\pi^{ot}_{freq} : E \rightarrow \mathbb{R}$ *where:*

- $V = \{a_1, ..., a_n\}$
- $E = \{(a_i, a_{i+1}) | 1 \leq i < n\}$
- *For* $(x, y) \in E$, $\pi^{ot}_{freq}(x, y) = |\{(a_1, a_2) \in E \mid (a_1, a_2) = (x, y)\}|$

The graph for trace $\sigma = \langle a, b, c, d, a, b, d \rangle$ is presented in Figure 4. For each object, we calculate the trace graph and for each node in the graph, we find centrality features. As an illustration, in Figure 4 the node list is $V = \{a, b, c, d\}$ and the corresponding *in-degree centrality* vector is $(1, 1, 1, 2)$. However, we need to assign a unique value to this object as the *in-degree centrality*. Thus, for each trace graph, the mean, variance, and standard deviation of all vector elements are calculated and inserted in the object attribute. For the mentioned vector (i.e., $(1, 1, 1, 2)$) the mean is 1.25, the variance is 0.25, and the standard deviation is 0.5. These values are added to the object attributes. For other features such as *closeness centrality*, we follow the same procedure. Using all these features, we enrich the object attributes. Now, using object attributes and object trace, we define object profile which is used as an input for clustering.

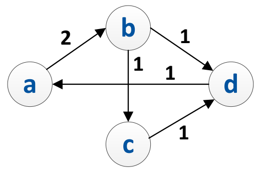


Figure 4: The graph of the trace trace$_{FL}(c)$.

Definition 8 (Object Profile)**.** *Let* $FL = (E^{ot}, \pi^{ot}_{act}, \pi^{ot}_{time}, \pi^{ot}_{case}, \leq^{ot})$ *be a flattened OCEL. We define object profile function for* $o \in O$ *where* $\pi_{otyp}(o) = ot$ *as* $op : U_o \rightarrow U^*_{act} \times U_{val} \times \ldots \times U_{val}$ *such that* $op(o) = (trace_{FL}(o), \pi_{ovmap}(o)(att_1), \ldots, \pi_{ovmap}(o)(att_n))$ *and* $att_1, \ldots, att_n \in dom(\pi_{ovmap}(o))$.

An example of the extracted profiles for batches is shown in Table 3. For example for b_1, the extracted trace, *treatment*, *workplace*, and *in−degree centrality mean* are the object attributes that constitute the b_1's profile. After constituting enriched object profiles, we apply clustering methods to the enriched object profiles.

5 CLUSTERING IN OCELs

Clustering results are affected by the distance measures that are used to measure the distance between object profiles. As Table 3 illustrates, an object profile consists of the control flow, numerical attribute values, and categorical attribute values. Levenshtein distance is used to measure the distance between the sets of activities of two control flows Bose and van der Aalst (2009). To calculate the distance between numerical values we apply Euclidian distance and String Boolean distance is employed to find the distance between categorical values. If the categorical values are the same the String Boolean distance is zero, otherwise the distance is one.

Using described distance metrics, we find the distance of the objects from each other to apply clustering algorithms. We have used the K-means algorithm and Hierarchical clustering to cluster similar objects. The K-means algorithm clusters data into k clusters based on minimizing within-cluster sum-of-squares criteria. The main techniques in hierarchical clustering are agglomerative and divisive approaches. In this paper, we have applied agglomerative clustering where smaller clusters of objects are combined to

Table 3: Object profiles extracted from an OCEL.

object ID	trace	treatment	workplace	...	in-degree centrality mean	in-degree centrality std	in-degree centrality var
b_1	$\langle print\ of\ production\ order, loading\rangle$	*painting*	*plant* 1	...	0.50	0.50	0.25
b_2	$\langle print\ of\ production\ order, ..., lubricate\rangle$	*polishing*	*plant* 1	...	1.00	0.00	0.00
b_3	$\langle loading, painting\rangle$	*painting*	*plant* 2	...	0.50	0.50	0.25

make a cluster. These clustering algorithms can be applied on the object profiles to create clusters of homogeneous objects. In the next section, we describe the transformation of the clustering results into an OCEL.

5.1 Transformation of the Clustering Results into OCEL

We should assign the clusters to the corresponding events to extract process models from the obtained clusters. Here, we propose *Existence* and *all* approaches. These approaches are comprehensively described in Section 1. In the *Existence* approach, we assign an event to the cluster, containing at least one object existing in that event. In the *All* approach, assuming we do clustering based on the objects with type *ot*, we assign an event to the cluster that contains all objects with the type *ot* that exist in that event.

By applying the clustering technique, we obtain sub-logs for each cluster. However, the aim is to apply clustering techniques to obtain less complex models. Thus, we define quality metrics on top of OC-DFGs.

5.2 Quality Metrics

To measure the quality of obtained models, we define some metrics. We first define the discovery of an OC-DFG which is the basis of the rest of the definitions.

Definition 9 (Discovery of an OCDFG). *Let* $L = (E, AN, AV, AT, OT, O, \pi_{typ}, \pi_{act}, \pi_{time}, \pi_{vmap}, \pi_{omap}, \pi_{otyp}, \pi_{ovmap}, \leq)$ *be an OCEL. Then we define* $OCDFG(L) = (A, OT, F, \pi_{freqn}, \pi_{freq})$ *where:*

- $A \subseteq U_{act}$ *is the set of activities.*
- $OT \subseteq U_{ot}$ *is the set of object types.*
- $F \subseteq ((\{\triangleright\} \cup A) \times (A \cup \{\square\})) \times OT$ *is the set of (typed) edges.*
- $\pi_{freqn} : A \not\rightarrow \mathbb{N}$ *is a node frequency measure.*
- $\pi_{freq} : F \not\rightarrow \mathbb{N}$ *is an edge frequency measure.*

OC-DFGs are one of the state-of-the-art object-centric models where each object type is shown with a specific color. In addition to the fitness criteria described in Berti and van der Aalst (2018), we define other measures such as size and density to find the complexity of the model. The smaller the graph, the simpler the structure is.

Definition 10 (Size). *Given an* $OCDFG = (A, OT, F, \pi_{freqn}, \pi_{freq})$, *we define the size of the* $OCDFG$ *as* $size(OCDFG) = |A| \times |F|$.

We have employed the graph density measure as a quality measure Lawler (2001). The more dense the graph, the more complex the model is.

Definition 11 (Density). *Given an* $OCDFG = (A, OT, F, \pi_{freqn}, \pi_{freq})$, *we define the density of the* $OCDFG$ *as* $density(OCDFG) = |A|/|F|$.

The size and density capture the general complexity information, however, to evaluate our approach we should compare the complexity of the obtained process models from clusters with the main process model. Therefore, we define the concepts related to improvements in size (i.e., C_sI) and density (i.e., C_dI) that are extensively explained in Ghahfarokhi et al. (2022). In the next section, we evaluate our approach on a real B2B process using the described evaluation metrics.

6 EVALUATION

To validate the proposed approach for object clustering in OCELs, we have performed a case study using the B2B dataset described in Section 2 representing a treatment process. This dataset contains 9004 events and three object types, namely *customer*, *order*, and *batch*. An *order* stands for a specific treatment to be applied to number of *batches* sent by a *customer*. The behavior of *customer* and *order* are similar, i.e., each *order* belongs to only one *customer*. Therefore, we evaluated our approach using *order* and *batch*. The process model of the whole process is a spaghetti-like model and too complex to interpret for the domain expert. Therefore, we applied the proposed clustering technique, described in Section 5, to discover simplified process models for each cluster. To find the optimal number of clusters we have employed Calinski-Harabasz, and Dendrogram for K-means and hierarchical clustering, respectively. The results confirm that at *batch*-level, three or four clusters, and at *order*-level, two or three clusters are the best choices. Considering the optimal number of clusters, we have applied Agglomerative and K-means clustering techniques to find the clusters. Both techniques were effective, nevertheless, the results of the K-means algorithm are more promising. By applying K-means

Table 4: Some characterizations of the main model.

The Main Model Properties				
No. of Nodes	No. of Edges	Fitness	Size	Density
25	118	0.83	2950	4.76

Table 5: The clustering result using K-means and *all* approach.

K-means									
Objects	No. of Clusters	No. of Nodes	No. of Edges	Fitness	Size	Density	Avg. Fitness	C_sI	C_dI
Order	2	24	106	0.85	2544	4.42	0.85	1.22	7.31
		12	34	0.89	408	2.83			
	3	24	106	0.85	2544	4.42	0.85	1.23	6.69
		12	27	0.86	324	2.25			
		10	20	0.98	200	2			
Batch	3	20	78	0.78	1560	3.9	0.82	2.38	43.74
		17	49	0.88	833	2.88			
		9	21	0.98	189	2.33			
	4	19	73	0.86	1387	3.84	0.88	3.27	44.95
		7	16	0.87	112	2.29			
		9	21	0.98	189	2.33			
		11	34	0.91	374	3.09			

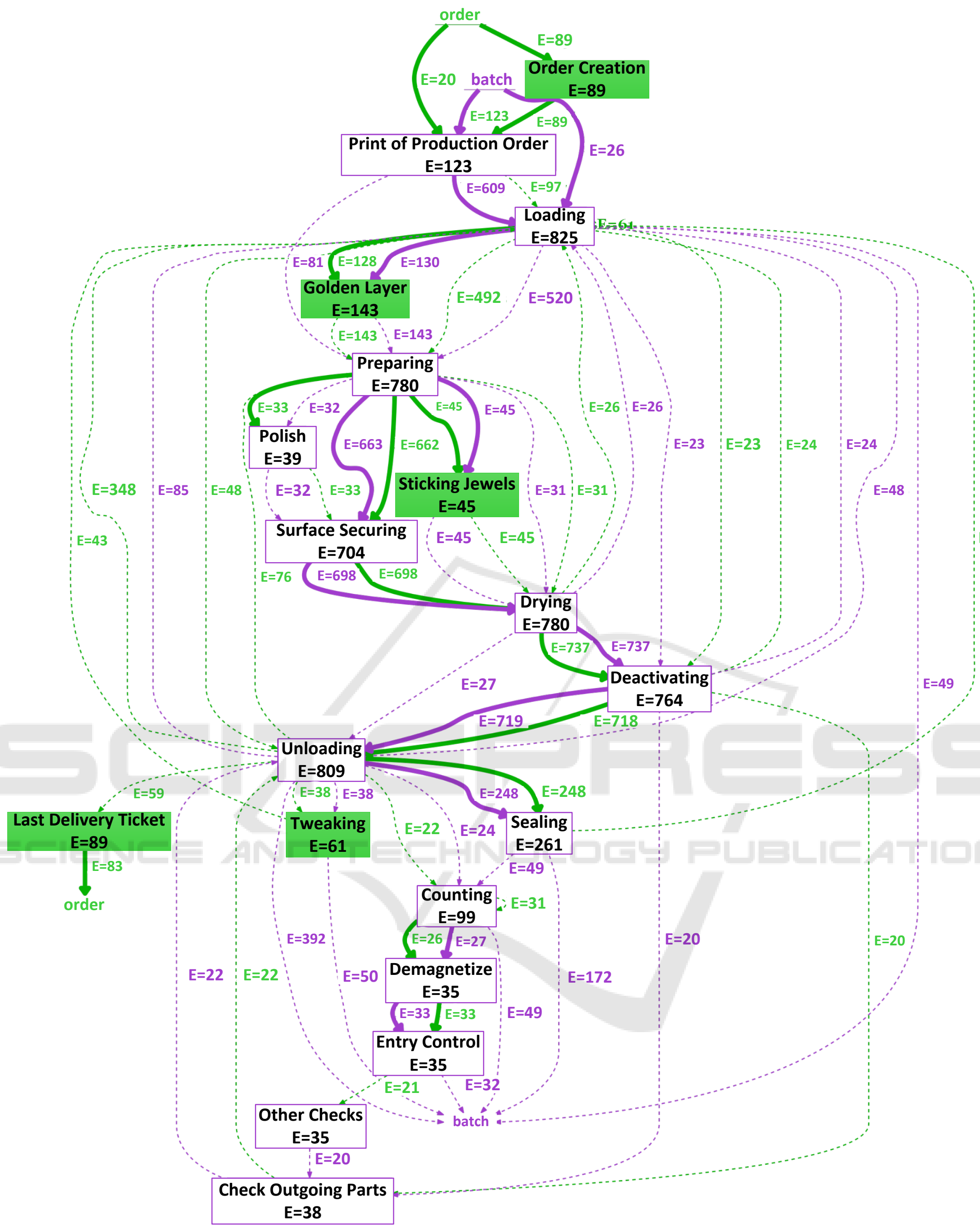
clustering on the set of object profiles, we got a set of objects in each cluster. Afterward, using *existence* and *all* approaches we managed to assign events to the clusters. Table 4 shows the characteristics of the main model and Table 5 reports the complexity and fitness of the respective models of the resulted clusters utilizing *all* approach. We evaluated the obtained process models using the fitness and complexity criteria described in Section 5.2. As the results show, the complexity of the obtained process models is reduced with the same or higher fitness. For example, the result of clustering based on batch with four clusters and *all* approach is shown in Figure 5. The results of using *existence* approach are provided in Ghahfarokhi et al. (2022). Besides the simplification of OC-DFGs per cluster show interesting points:

- In two clusters (i.e., Cluster 1, and Cluster 2) the process has started with *order creation*, however, in Cluster 3 there is no *order creation*. After discussion with the expert, we realized that Cluster 3 shows the rework process of the items that experienced failures in their previous treatment process. Therefore, no *order creation* is executed in these processes.
- There is a difference between Cluster 1 and Cluster 2. *Print of order production* is followed by *hanging pieces* in Cluster 2 whereas it is followed by *loading*, in Cluster 1. We recognized that the process, shown in Cluster 2, refers to small items such as nuts and bolts. Therefore, we hang them to plate both sides of them. However, cluster 1 represents the process of larger items such as bottles that we should load to do the treatment.
- The *last delivery ticket* activity shown in Cluster 1 shows the delivery status. When an employee finishes an order which is divided into several batches, the shipping process starts. Each delivery in the shipping process requires a delivery ticket. The *Last delivery ticket* refers to the last shipment of an order and its delivery ticket.

We have shown the process models of three clusters in Figure 5. The model of the remained cluster is shown and discussed in Ghahfarokhi et al. (2022). As can be seen, the proposed technique distinguishes different processes successfully. To sum up, we have applied the proposed clustering technique on a B2B process to derive simpler models. The obtained process models are simplified and meaningful models that help the user gain insights into the process.

7 CONCLUSION

Process mining techniques provide valuable insights about process executions, however, most of the process mining techniques, focus on event logs with a single case notion. In reality, there exist processes with multiple interacting objects which are investigated in a new branch of process mining called object-centric process mining. Several process discovery techniques such as Object-Centric Directly Follows Graphs (OC-DFGs) discovery have been developed to discover process models from object-centric processes, but the discovered process models usually suffer from complexity. In this paper, we propose two approaches (i.e., *all* and *existence*) to obtain meaningful process models by clustering objects in an object-centric event log (OCEL). Furthermore, we enriched the OCEL with some graph-related features and introduced complexity measures to evaluate the quality of OC-DFG models. We have applied our approach on a real-life B2B log of a manufacturing company applying surface treatment operations, e.g., lubricating and polishing. The results are promising where



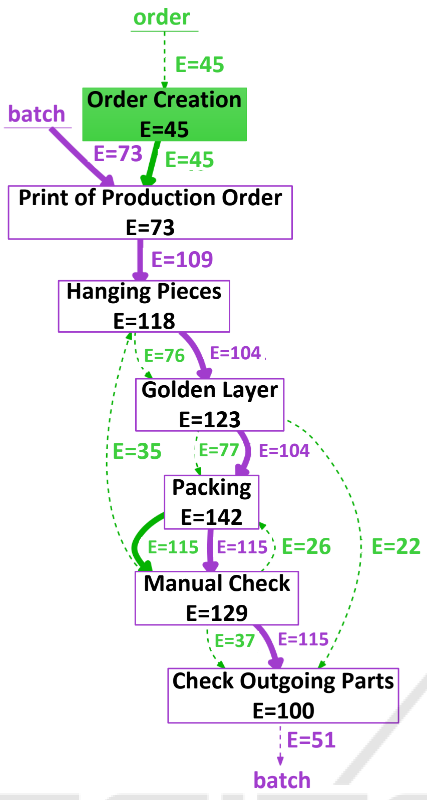
(a) Cluster 1.

Figure 5: Final result of the clustering of batch based on *all* approach.

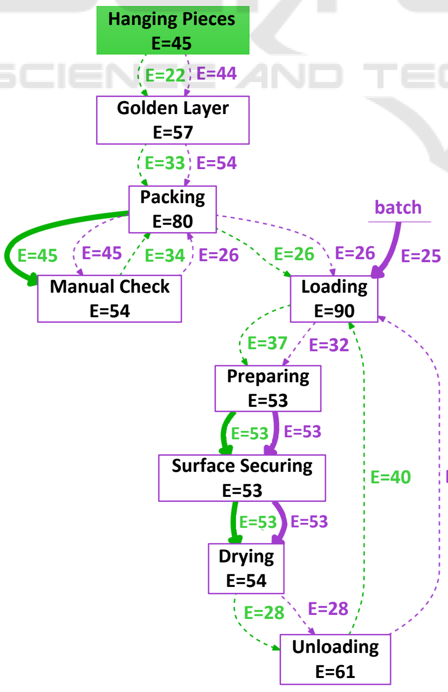
discovered process models distinguish the process of different item types. For future work, we aim to evaluate the proposed approach on additional real data sets and use various quality metrics to evaluate the quality of the obtained process models more precisely.

ACKNOWLEDGMENTS

We thank the Alexander von Humboldt (AvH) Stiftung for supporting our research. Funded by the Deutsche Forschungsgemeinschaft (DFG, Ger-



(b) Cluster 2.



(c) Cluster 3.

Figure 5: Result of the clustering based on *all* approach (cont.).

man Research Foundation) under Germany's Excellence Strategy–EXC-2023 Internet of Production – 390621612.

REFERENCES

Bender, E. and Williamson, S. ((2010)). *Lists, decisions and graphs*. San Diego: University of California at San Diego.

Berti, A. (2022). Filtering and sampling object-centric event logs. *arXiv preprint arXiv:2205.01428*.

Berti, A., Ghahfarokhi, A. F., Park, G., and van der Aalst, W. (2022). A scalable database for the storage of object-centric event logs. *arXiv preprint arXiv:2202.05639*.

Berti, A. and van der Aalst, W. (2018). Extracting multiple viewpoint models from relational databases. In *SIMPDA*, pages 24–51. Springer.

Bose, R. and van der Aalst, W. (2009). Context aware trace clustering: Towards improving process mining results. In *SIAM*, pages 401–412. SIAM.

Cohn, D. and Hull, R. (2009). Business artifacts: A data-centric approach to modeling business operations and processes. *IEEE Data Eng. Bull.*, 32(3):3–9.

Esser, S. and Fahland, D. (2021). Multi-dimensional event data in graph databases. *Journal on Data Semantics*, 10(1):109–141.

Faria Junior, E. R., Neubauer, T. R., Fantinato, M., and Peres, S. M. (2023). Clustering analysis and frequent pattern mining for process profile analysis: An exploratory study for object-centric event logs. In *ICPM*, pages 269–281. Springer.

Ghahfarokhi, A. F., Akoochekian, F., Zandkarimi, F., and van der Aalst, W. (2022). Clustering object-centric event logs. *arXiv preprint arXiv:2207.12764*.

Ghahfarokhi, A. F., Berti, A., and van der Aalst, W. (2021a). Process comparison using object-centric process cubes. *arXiv preprint arXiv:2103.07184*.

Ghahfarokhi, A. F., Park, G., Berti, A., and van der Aalst, W. (2021b). OCEL: A standard for object-centric event logs. In *SIMPDA*, pages 169–175. Springer.

Ghahfarokhi, A. F. and van der Aalst, W. M. (2021). A python tool for object-centric process mining comparison. In *Proc: ICPM Doctoral Consortium Demo Track*, pages 31–32. Springer.

Jalali, A. (2022). Object type clustering using markov directly-follow multigraph in object-centric process mining. *IEEE Access*, 10:126569–126579.

Lawler, E. (2001). *Combinatorial optimization: Networks and matroids*. Courier Corporation.

Lu, X., Nagelkerke, M., Van De Wiel, D., and Fahland, D. (2015). Discovering interacting artifacts from ERP systems. *IEEE Transactions on Services Computing*, 8(6):861–873.

Meroni, G., Baresi, L., Montali, M., and Plebani, P. (2018). Multi-party business process compliance monitoring through IoT-enabled artifacts. *Information Systems*, 73:61–78.

Van der Aalst, W. (2016). Data science in action. In *Process mining*, pages 3–23. Springer.