Business Object-centric Trace Clustering

Bachelor Thesis

presented by
Constantin Hannes Ernstberger
Matriculation Number 1645394

submitted to the
Data and Web Science Group
Prof. Dr. Han van der Aa
University of Mannheim

September 2021

Abstract

While Process Mining has proven itself as a valuable tool to deliver insightful analyses of operational processes, it remains to struggle when working with unstructured processes often resulting in 'spaghetti-like' models. Therefore, complexity-reducing measures are necessary to improve the quality of results. A commonly used technique is trace clustering, a divide-and-conquer approach, which splits the event log into homogeneous subsets. While reducing the size of the event log is enough to improve process mining results for some techniques, more advanced techniques also require semantic information. After extracting this semantic information from the event log, using novel approaches like the role labeling by Rebmann and van der Aa, it enables techniques such as object-centric process mining, the investigation of behavior, and inter-relationships associated with specific business objects. An object-centric approach enables the possibility to analyze complex event logs from social media, discover novel models such as the object-centric behavioral constraint models (OCBC) or object-centric Petri nets. Nevertheless, object-centric process mining faces the same problems as traditional techniques when it comes to the immense sizes of complex and unstructured event logs. Since these novel techniques do not treat data as sequences of abstract events, which traditional trace clustering approaches use to find similarities between traces, grouping traces based on traditional techniques would not provide the necessary results. Therefore, in this paper, we present an approach to consider semantic features in proven distance measures, in order to cluster traces in an object-centric manner. We extend the Levenshtein distance and the Euclidean distance, so the relationship between business objects and actions is considered. Using eight real-life event logs, and their extracted semantic information, our approaches' usefulness is demonstrated.

Contents

List of Algorithms

List of Figures

List of Tables

Chapter 1

Introduction

Data Mining is the extraction of implicit available, but nontrivial and useful knowledge from large, dynamic, and often complex datasets. Since the early nineties, data mining has grown in importance, due to the increasing number and size of databases. The immense mass of stored data has become increasingly difficult to analyze with conventional methods. Often data is stored in nominal as well as metric attributes, making statistical approaches less appropriate. Therefore, various new disciplines emerged to complement the field of statistics [3].

In the field of operational processes, systems as Customer Relationship Management or Enterprise Resource Planning do not use spreadsheet data. Instead, the data relates to processes, a collection of continuous execution of activities, so-called event data. To discover, monitor, and improve these real processes, a family of a-posteriori analysis techniques emerged, process mining. It provides the ability to close the gap between traditional and process-oriented data analysis methods. The only requirement and starting point of any process mining endeavor is the event log, where each event refers to an activity or a well-defined step in a process instance, so-called trace [19]. Trace clustering applies the ideas of traditional clustering in data mining to event data. Based on the abstract nature of events within traces, trace clustering measures the similarity between different traces to group similar ones together. The resulting process models of each cluster are no longer incomprehensible improving the quality of the process mining endeavor.

As mentioned, these traces are treated as sequences of abstract events. However, more advanced process mining techniques consider specific kinds of information in these events, in order to avoid neglecting certain facets of reality. Object-centric Process mining suggests that novel techniques should take this information and inter-relationships into account. Since this necessary semantic information is regularly not readily available in event logs, and only recent approaches, as the role

labeling by Rebmann and van der Aa [13], started to extract this information, trace clustering has not yet incorporated these semantic features. Therefore, taking the object-centric approach to trace clustering and considering semantic information, might not only provide the possibility to improve object-centric process mining results, but to improve the clustering itself.

1.1 Problem Statement

While some techniques take the sequence of events within a trace into consideration, when it comes to the activity itself, current techniques only consider the abstract, textual activity of events within a trace. Resultantly, considering only this one attribute, all similarity measures are one-dimensional. In order to take an object-centric approach, we need to take at least two attributes into consideration: business objects and the corresponding actions. Not only do we need to consider two separate attributes, but we also need to respect the relationship between them since both are part of a specific event.

To illustrate the potential of object-centric trace clustering, consider the three simplified traces in table 1.1. On the one side, trace 1 shows a successful withdrawal of cash, while trace 2 shows a failed withdrawal. On the other side, trace 3 shows a successful transfer of money. Therefore, trace 1 and trace 2 represent the process of withdrawing money, while trace 3 represents the process of transferring money. With first intuition, trace 1 and trace 2 should be more similar than trace 1 and 3 or trace 2 and 3. Nevertheless, trace 1 and trace 2 as well as trace 1 and trace 3 both have a difference of three activities. Therefore, there is no difference in similarity.

Trace 1	**Trace 2**	**Trace 3**
Activity	Activity	Activity
insert card	insert card	insert card
select withdrawal	select withdrawal	select transfer
insert pin	insert pin	insert pin
select amount	insert pin	select amount
complete withdrawal	insert pin	complete transfer
take cash	cancel withdrawal	take receipt
take card	take card	take card

Table 1.1: A simplified event log displaying regular activities

However, if we take an object-centric approach, as shown in table 1.2, and consider the business objects of all traces, trace 1 and trace 2 differ in only two

objects, while trace 1 and trace 3 differ in three. Resultantly, trace 1 and trace 2 are considered more similar and would result in one cluster representing withdrawal processes.

Trace 1		**Trace 2**		**Trace 3**	
Object	Action	Object	Action	Object	Action
card	insert	card	insert	card	insert
withdrawal	select	withdrawal	select	transfer	select
pin	insert	pin	insert	pin	insert
amount	select	pin	insert	amount	select
withdrawal	complete	pin	insert	transfer	complete
cash	take	withdrawal	cancel	receipt	take
card	take	card	take	card	take

Table 1.2: A simplified event log displaying semantic roles

1.2 Contribution

In this thesis, a novel object-centric trace clustering approach based on proven distance measures is proposed. More precisely, this thesis proposes two similarity measures for object-centric trace clustering based on existing trace clustering techniques: For our trace syntax-based approach, we are adjusting the Levenshtein distance, and for our feature-vector-based approach, we are adjusting the Euclidean distance. Using k-means clustering with the computed distances, we will evaluate the resulting clusters based on internal validation against the traditional distance measures. Additionally, we will analyze the resulting process models based on fitness, precision, generalization, and simplicity. The effect of exploiting semantic information is evaluated using internal validation as well as process model analysis. Since both novel approaches are compared to their traditional form, advantages and possible disadvantage can be seen.

Moreover, researchers can use this object-centric approach as a framework. Not only input parameters, as well as event logs, can be modified, but researchers can add more clustering algorithms to test the novel distance measures even further. Additionally, other evaluation metrics or process model quality criteria can be applied to adopt results to other requirements. Finally, researchers can leave the object-centric approach and extend the similarity measure by the passive and active resources to exploit all four semantic roles of the augmented event logs.

1.3 Thesis Structure

In Chapter 2, we start with the preliminaries and background information necessary for our approach. These include an introduction to process mining and the event log and the transition from data clustering to trace clustering. Afterward, we present related work for similarity measures in trace clustering, which is followed by a closer look into Rebmann and van der Aa's semantic labeling approach and other work in the area of object-centric process mining. Then Chapter 3 introduces our business object-centric trace clustering approach, which includes two distance measures, the augmented Levenshtein distance, and the augmented Euclidean distance. We explain the whole clustering process starting with the input and data preprocessing which is followed by the steps of k-means clustering and the final output. Before we start evaluating using internal validation and process model analysis, we present the implementation of our approach in Chapter 4. In Chapter 5 we evaluate our approach using internal validation as well as process model evaluation with multiple real-life data sets from different areas of application. Finally, we end this thesis with a conclusion in Chapter 6 as well as a look into a potential future work in Chapter 7.

Chapter 2

Background and Related Work

In this chapter, relevant concepts and notions that are used in this thesis are explained. Starting with process mining and the event log, we will transition to data and trace clustering. Additionally, existing trace clustering approaches are investigated that will play an important role in the latter of this thesis. One is the feature vector-based similarity, which maps traces as vectors, in order to calculate the distance between them. The other is the syntax-based similarity, which introduces the syntax similarity. Finally, semantic labeling and information will be investigated by giving an introduction to the semantic information extraction approach by Rebmann and van der Aa [13], which is followed by a first look into object-centric process mining.

2.1 Process Mining

Process mining describes a family of a-posteriori analysis techniques. The main idea is to discover, monitor, and improve real processes with extracted process

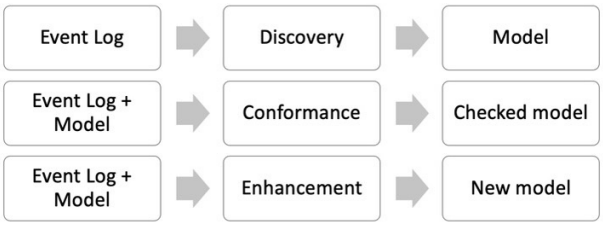

Figure 2.1: The three main domains of process mining.

knowledge based on event log data. While traditional data mining techniques do not take the end-to-end nature of process data into account, process mining incorporates both process model-driven and data mining approaches.

There are three main domains within process mining: discovery, conformance checking, and enhancement. The first and most prominent technique is discovery. Derived information from the event log is transformed into well-known notations, as BPMN models or Petri nets, in order to reproduce the observed behavior. In conformance checking, the event log is used to check if reality conforms to a specific model. Using predefined rules, deviations are identified, explained, and measured based on their severity. Additionally, it can be used to measure the performance of process discovery algorithms. Finally, enhancement aims to change or improve an a-priori model. It offers two activities: repair, as in modifying the model to better reflect reality, and extension, as in adding a new perspective to the model [20] [21] [11].

2.2 Event Log

Case Id	Activity	Timestamp	...	Ressource	State
1	Register Request	22:07	...	System A	started
1	Register Request	22:07	...	System A	completed
2	Inform customer	22:30	...	System B	started
2	Inform customer	22:32	...	System B	completed
1	Work on Request	22:40	...	User 1	started
1	Work on Request	22:49	...	User 1	completed
...	...	...	...	...	...

Table 2.1: Example event log

The starting point for any process mining endeavor is the event log. Systems that produce such event logs are Customer Relationship Management (CRM) or Enterprise-Resource Planning (ERP) systems. Typically, those also include very detailed information about the executed activities, including the performer of the event, the timestamp of the event, or recorded data elements. Whenever possible, process mining techniques use this extra information. Process mining techniques assume that events are recorded sequentially such that each event refers to an activity and is related to a specific case. A sequence of events, representing an orderly of the same process instance from the start to the end, is referred to as a trace. Resultantly, each process instance is manifested as a trace, while an event log is

represented by a set of multiple traces [20].

Table 2.1 shows a fragment of an event log. Every row outlines an event and each column represents an attribute. Only the first three columns, Case Id, Activity, and Timestamp, are essential requirements for an event log. The Case Id column indicates which process instance an event belongs to. The Activity column is used for the event's activity name, while the Timestamp column represents the completion time of the corresponding activity. The remaining two columns simply provide additional information that could be missing in other event logs.

2.3 Data Clustering

Clustering is an unsupervised data mining technique. It is a process of partitioning unlabeled data with the goal of discovering natural groups of a set of patterns, points, or objects. Objects a grouped based on their similarity. Resultantly, objects within a cluster are similar to each other. In contrast, they have minimum similarity to objects of other clusters. To separate data into groups, there are many different clustering algorithms.

Partition clustering algorithms are based on specifying an initial number, k, of groups and iteratively reallocating objects among non-overlapping groups. Typically, all clusters are determined at once. Two of the most used and among the best performing partitioning algorithms are the greedy approaches k-means and k-medoids.

Since partitional clustering algorithms find all clusters simultaneously, they do not impose a hierarchical structure. In contrast, hierarchical clustering methods find nested clusters. A bottom-up method is agglomerative clustering. Starting with a single point as its own cluster, most similar pairs of traces are merged successively until a final cluster is obtained. Using a top-down method, in divisive clustering all points are contained in one cluster. Recursively, clusters are divided into smaller clusters.

Next to the most prominent clustering algorithms, there are also other approaches. The density-based clustering method is based on the fact that within each cluster there is a typical density of points. This density is much higher than outside this cluster. These points are recognized as noise. One highly scalable technique is grid-based clustering. It builds a space with a finite number of cells forming a grid. With a new object space inserted into the grid, the clustering process itself is done in this space [12] [16].

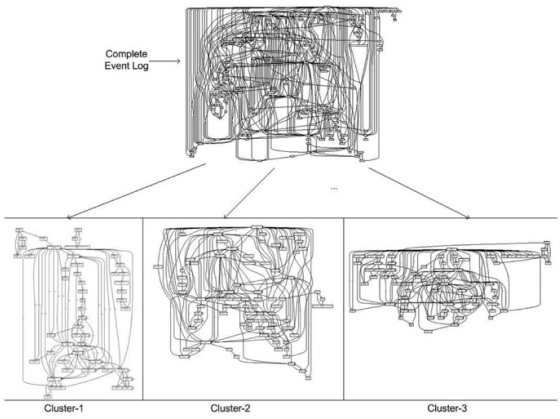

Figure 2.2: The effect of trace clustering in process mining [4].

2.4 Trace Clustering

Since discovering a single model from a complex event log often results in incomprehensible, so-called 'spaghetti-like', models, complexity-reducing measures have to be used. Similar to data clustering, trace clustering groups process data based on similarity and splits the event log into homogeneous subsets. Consequently, models discovered from smaller logs are less complicated and easier to read.

There are three different approaches for measuring the similarity between traces: The feature-vector-based transforms traces into vectors. Trace-sequence-based clustering introduces a syntax similarity between two traces. Finally, model-based trace clustering shifts the definition of similarity towards the quality of the discovered models [15] [24]. However, in this thesis, the focus lies on the following two similarity measures.

2.4.1 Feature-vector-based

Feature-vector-based similarity follows the ideas of traditional data clustering. One of the most used approaches to transforming a trace into a vector is the so-called bag-of-activities approach. Each dimension of the vector corresponds to an activity. The number of unique activities defines the dimension of the vector. Each value of a vector corresponds to the frequency of the activities within the trace. Using standard distance metrics, as the Euclidean distance, Hamming distance, or Jaccard distance, the similarity between two traces can then be estimated. This approach

is then deployed using distance-based algorithms, such as k-means, quality threshold, or agglomerative hierarchical clustering [15]. However, this approach has two drawbacks. First, it does not provide context information, as it does not capture the dynamics of process execution. Second, it ignores the order of execution [5].

The k-gram model incorporates context into the vector space model by considering subsequences of events, which also capture the order of execution. A subsequence of k activities is referred to as k-gram. A trace **abacaab** in 2-gram corresponds to **{ab,ba,ac,ca,aa}** and in 3-gram to **{aba,bac,aca,caa,aab}**. Such traces can then be transformed into a vector. However, the size of this model increases drastically in size with an increasing number of activities and value of k. It also incurs a huge computational overhead and selecting an ideal k is also non-trivial [5].

Another approach to constitute feature vectors is using frequent (sub)sequences of activities. It was one of the first approaches to incorporate trace clustering into process discovery [7]. Lu et al. [24] used frequent sequence patterns to group patients and discover process maps. A frequent sequence pattern is a sequence of events that frequently occurs in traces and occurs more often than a specific threshold.

2.4.2 Trace-sequence-based

Trace-sequence-based clustering proposes that the syntax similarity between two traces can be measured. Therefore, traces are compared based on their syntactic differences. One fundamental measure is the Levenshtein distance, also called edit distance. The Levenshtein distance is defined as the minimum number of edit operations necessary in order to transform one trace into another. Possible operations are insertions, deletions, or substitutions [9].

In [5], the similarity between traces was also measured using the General Edit Distance. However, they applied a cost function with different characteristics. Scores for substitutions and insertions/deletions of activities are derived for similarity. The basic idea is to compare the actual frequency of a pair of activities to their expected frequency if occurring independently.

2.5 Semantic Labeling

In traditional process mining, traces are treated as sequences of abstract symbols. In contrast, advanced process mining techniques, such as social network analysis, consider specific kinds of information, so-called semantic components, contained in events' labels or attributes. However, these semantic components are not readily

availably in most event logs, due to a lack of standardization of attributes in event logs. Instead, the relevant information is often captured as part of unstructured, textual data attributes. Resultantly, the necessary information cannot be exploited by the process mining process. Therefore, Rebmann and van der Aa [13] proposed a novel approach for extraction semantic information from unstructured text. The approach consists of three separate steps. First, textual attributes are identified using natural language processing techniques. Second, extracted textual attributes are labeled on instance-level. Third, the semantic roles of miscellaneous attributes are labeled on instance-level. During this process, four main categories of commonly observed semantic roles in event logs are targeted: business objects, actions, actors, and passive resources. Business objects are broadly referred to as the main object relevant to an event. Actions are applied to business objects. Actors are the active resource in an event, while passive resources are the passive part involved in an event, often as recipients. These four semantic roles enable a broad range of fine-granular insights into the execution of a process. Especially the roles business object and action provide the ability to obtain insights into the object moving through a process. Even further, inter-relations and life cycles can be investigated. The analysis that focuses on these two roles is called object-centric analysis. It enables the possibility to investigate the behavior associated with specific business objects.

Case Id	Activity	Timestamp	...	Business object	Action
1	Register Request	22:07	...	request	register
1	Register Request	22:07	...	request	register
2	Inform customer	22:30	...	customer	inform
2	Inform customer	22:32	...	customer	inform
1	Work on Request	22:40	...	request	work on
1	Work on Request	22:49	...	request	work on
...	...	...	...	...	...

Table 2.2: Example of an augmented event log displaying semantic roles

2.6 Object-centric Process Mining

Process mining works with attributes case identifier, activity name, time stamp, and possibly additional ones stored in the event log. Often, there are multiple candidate identifiers leading to different views on the same process. Therefore, no single case notation exists, since any would neglect certain facets of reality. This is the main argument of Object-centric Process Mining, suggesting that there are ”multiple in-

tertwined case notions". Resultantly, process mining techniques should take these inter-relationships into account [17]. Van der Aalst introduced a formal definition for event logs to tackle this problem where events contain multi-valued entity identifiers, while also introducing two terms relevant to Object-centric process mining: convergence and divergence. The first focuses on one event that may be related to different cases. The latter looks at multiple instances of the same activity within a case [17].

In order to analyze complex event data from social media, as users' behavior patterns or operational processes in media sites, which contain one-to-many and many-to-many relations, Li et al. [10] propose a novel type of process mining techniques, based on object-centric behavioral constraint models (OCBC). An "OCBC model combines data models with a declarative behavioral perspective."

From an object-centric event log, Berti et al. [18] discover Petri nets with places that correspond to object types and transitions with possible collections of objects of different types. These object-centric nets visualize the complex relationships among objects of different types.

2.7 Discussion

As mentioned in the corresponding sections, both trace syntax-based approaches, as well as feature vector-based approaches, have their drawbacks. However, all of them have proven themselves as valuable tools for improving process mining results. With the new approach to process mining which focuses on semantic features, in order to tackle the analysis of other areas, which generate a lot of data, the question arises if these can also be used for improving trace clustering results.

The main idea of this thesis is to take an object-centric approach to trace clustering. Since both traditional trace clustering, as well as the object-centric approach to process mining, have proven their importance, combining both could provide great possibilities.

Chapter 3

Business Object-centric Trace Clustering

This chapter discusses the object-centric trace clustering approach proposed in this thesis. The first step in the clustering process is identifying the similarity between two traces. For our object-centric approach, we are adjusting two different, proven techniques to identify the similarity between two traces. In trace-sequence-based clustering, the Levenshtein distance expresses the number of edits necessary to transform one trace into another trace. Edits include insertions, removals, or replacements of events. Following the ideas of traditional data clustering, the feature-vector-based similarity is measured by transforming traces into vectors and computing the distance between them.

3.1 Overview

Input for our business object-centric trace clustering approach is labeled event data. This event data is filtered based on trace variants, in order to avoid duplicates. Considered as duplicates are traces with the same control-flow perspective or sequence of events.

In the next step, traces need to be **preprocessed**, so we can apply our distance metrics. This step is different for our two approaches since both follow different trace clustering techniques. For the augmented Levenshtein distance, traces need to be transformed into lists of business objects and actions in correct order. For the augmented Euclidean distance, traces need to be transformed into multiple vectors.

The parts of the process of **clustering** traces using k-means clustering are the following for both distance measures:

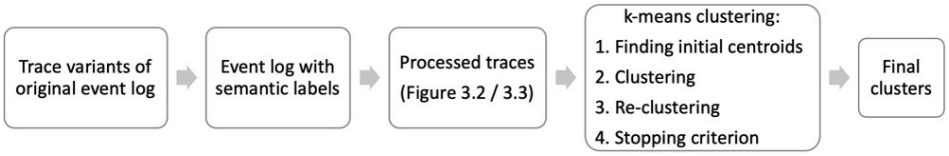

Figure 3.1: Overview of the proposed business object-centric trace clustering approach

1. **Finding initial centroids:** Initial centroids are selected randomly or selectively based on minimum distance to each other.
2. **Clustering:** The distances between the centroids and all traces are computed. Clusters are assigned to the centroid with the closest distance.
3. **Re-clustering:** Based on the traces within a cluster, the new centroid is calculated. Now step 2 is repeated.
4. **Stopping criterion:** Step 3 is repeated until the centroids do not change or eight iterations were completed.

As the **output** we receive labeled traces indicating the according to the cluster.

3.2 Augmented Levenshtein Distance

For our trace-syntax-based approach, we are transforming the Levenshtein distance. Traditionally, this is the minimum edit distance of events between two traces, with each edit increasing the distance between two traces by one. However, in our object-centric approach, we look at the business objects and the actions of each event separately.

As seen in figure 3.2, each trace is represented by two lists. The first list includes all business objects addressed within the trace's events in the correct order. The same goes for the actions, which are listed in the second list. Based on the Wagner-Fischer algorithm, which is based on the observation that if a matrix is reserved to hold the Levenshtein distances between all prefixes of the first string and all prefixes of the second string, then it is possible to compute the values in the matrix using bottom-up dynamic programming. The final distance is then the last value computed [22]. However, to compute the augmented Levenshtein distance between two traces, the Wagner-Fischer-Algorithm is adjusted slightly.

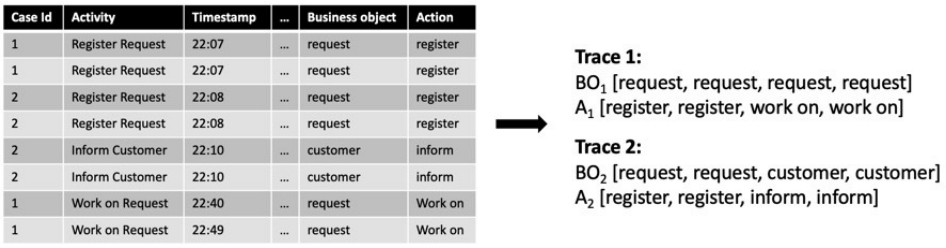

Figure 3.2: Transform traces into business object list and action list.

Algorithm 1

AUGMENTEDLEVENSHTEIN(BO_1, BO_2, A_1, A_2)

$matrix \leftarrow zeros[len(BO_1), len(BO_2)]$
for $i \leftarrow 0$ to $len(BO_1)$ **do**
  $matrix[i, 0] \leftarrow i$
end for
for $j \leftarrow 0$ to $len(BO_2)$ **do**
  $matrix[0, j] \leftarrow j$
end for
for $i \leftarrow 1$ to $len(BO_1)$ **do**
  for $j \leftarrow 1$ to $len(BO_2)$ **do**
    if $BO_1[i-1] = BO_2[j-1]$ **then**
      if $A_1[i-1] \neq B_2[j-1]$ **then**
        $substitutionCost \leftarrow 0.5$
      else
        $substitutionCost \leftarrow 0$
      end if
    else
      $substitutionCost \leftarrow 1$
    end if
    $matrix[i, j] \leftarrow min(matrix[i, j-1] + 1, //insertion$
    $matrix[i-1, j] + 1, //deletion$
    $matrix[i-1, j-1] + substitutionCost)//substitution$
  end for
end for
return $matrix[len(BO_1), len(BO_2)]$

			payment	permit	permit	permit
			submitted	approved	approved	approved
		0	1	2	3	4
payment	submitted	1	**0**	1	2	3
permit	rejected	2	1	**0.5**	1.5	2.5
request	submitted	3	2	1.5	**1.5**	2.5
payment	approved	4	3	2.5	2.5	**2.5**

Figure 3.3: One Example of a resulting matrix for computing the augmented Levenshtein distance between two traces.

We use figure 3.3 as a running example, in order to explain algorithm 1. The figure displays the resulting matrix of computing the distance between two traces. The first trace consists of the business objects **payment, permit, request, payment** and the actions **submitted, rejected, submitted, approved**. The second trace consists of the business objects **payment, permit, permit, permit** and the actions **submitted, approved, approved, approved**. As both traces have a length of four, we build a quadratic matrix from 0 to 4 and set all values to zero. Now we set the values of row 0 and column 0 to the index of the business object/action in their corresponding list, so payment — submitted is set to 1, permit — rejected is set to 2, and so on. The matrix is now prepared and we can start with the process of comparing the two traces, which is represented by the for loop from line 8 to line 23. Hovering over the bold and colored diagonal of numbers reveals the operations performed:

- **Green:** Both the business objects as well as the actions are identical. Therefore, the distance remains **0**.
- **Blue:** Even though the business objects are identical, the actions are different. Resultantly, the distance is only increased by **0.5**.
- **Orange:** The business objects are different, resulting in a distance increase of **1**.
- **Yellow:** The actions might be identical, however, this does not influence the distance if the business objects are different. Distance is again increased by **1**.

This distribution of weight between the business objects and actions puts more importance on the business objects. This decision was taken, due to the fact that business objects are the main elements relevant to an event. Additional operations

would be the simple insertion or deletion of an event, both increasing the distance by 1 since they are also addressing business objects.

Since the yellow field is also in the bottom-right, it represents the final augmented Levenshtein distance between the two traces.

3.3 Augmented Euclidean Distance

Our feature-vector-based approach is the augmented Euclidean distance. In order to apply the Euclidean distance to trace clustering, traces have to be transformed into vectors. Based on the bags-of-activities approach, all traces are represented by n-dimensional vectors, representing the frequency of the n possible events within the event log. However, in our object-centric analysis, we are working with semantic features of events. Therefore, we adopted the bags-of-activities approach to the augmented event log.

In order to take an object-centric approach, a trace cannot simply be represented by one vector, instead, each trace will be represented by n 'business object' vectors. First, all n unique business objects and m possible actions within the event log are identified. To build the n 'business object vectors' for each trace, we count the frequency of all m possible actions considering a specific object. Resultantly, each trace will be represented by n m-dimensional vectors, with each vector representing the frequency of actions of a specific object, as seen in figure 3.3.

When measuring the similarity between two traces, each business object will now be inspected separately with its according actions. Therefore, the Euclidean distance will be computed for each 'business object' vector pair, which will leave us with n different distances. The sum of all n Euclidean distances will then be the final distance between two traces.

This calculation process can be seen in algorithm 2: First, the distance is set to 0. Then we iterate through all business objects within the event log. For each object, we select the according business object vector. Now we compute the Euclidean distance between these two vectors and add it to the overall distance. The equation of the Euclidean distance can be seen in (3.1) [15]. After doing so for all business objects, we receive the final distance.

$$EuclideanDistance(c_j, c_k) = \sum_{l=1}^{n} \sqrt{|j_{jl} - i_{kl}|^2} \tag{3.1}$$

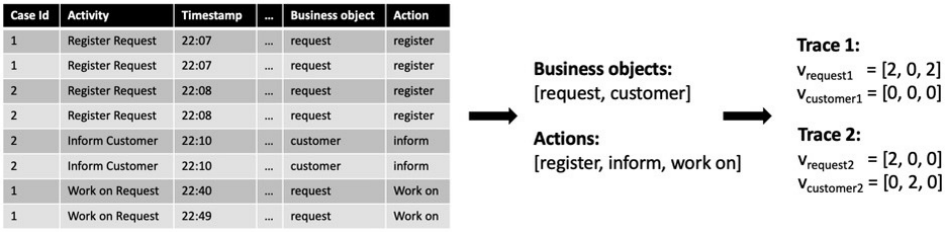

Figure 3.4: Transform traces into 'business object vectors'.

Algorithm 2

AUGMENTEDEUCLIDEAN($trace_1$, $trace_2$, $objects$)

1: $distance \leftarrow 0$
2: **for** `obj in objects` **do**
3: $v_{obj_1} \leftarrow trace_1[obj]$
4: $v_{obj_2} \leftarrow trace_2[obj]$
5: $distance \leftarrow distance + EuclideanDistance(v_{obj_1}, v_{obj_2})$
6: **end for**
7: **return** $distance$

3.4 Clustering

For the process of clustering traces, we are implementing k-means clustering, the most commonly known used in practice among partitioning methods. It partitions the data into k groups while constructing k clusters, in a computationally effective manner [15]. Traditionally, the initial centroids are selected randomly, which allows the possibility for choosing identical or extremely similar traces as centroids. To avoid this issue, we also implemented an advanced initialization technique. This technique is inspired by the thought process of pillars' placement to build a stable house. The pillars need to be placed as far as possible from each other, in order to withstand the pressure of a roof [1]. Using this idea, our technique selects initial centroids based on a minimum distance to all other centroids.

After each iteration, a new centroid for each of the clusters needs to be computed. As the name proposes, this centroids is the mean of all traces within the cluster. In the case of the augmented Euclidean distance computing, the mean is trivial. We are simply calculating the average vector for each business object. Doing the same using textual attributes is rather difficult. Therefore, we needed a different approach for the Levenshtein distance. A trace is now chosen as a new centroid if it has the lowest average distance to all other traces within the cluster.

Chapter 4

Implementation

This chapter discusses the implementation of our business object-centric trace clustering approach. The algorithm used for this object-centric approach is implemented using the Python programming language and is publicly available at Github. First, all libraries that are used in this implementation are introduced briefly. Then, the choice of the k-means clustering algorithm is explained, while we are also looking at the architecture of the implementation.

- **PM4PY** is an open-source, process mining library for Python. It provides object management of the event log, discovery as well as conformance checking algorithms, and process model visualization [2] Next to converting event logs into pandas data frames, pm4py is used in this thesis in order to filter event logs based on trace variants. Additionally, the three process discovery algorithms Alpha Miner, Inductive Miner, and Heuristic Miner are borrowed for obtaining Petri Nets.

- **Pandas** is an open-source that provides fast, powerful, and flexible data analysis and manipulation. In this thesis, event logs are stored on Pandas data frames for working with process data.

- **NumPy** is an open-source library that supports the use of high-level mathematical functions and multi-dimensional matrices. In this thesis, it is used for calculating the Euclidean distances between two vectors and storing values in matrices.

- **Scikit-learn** is an open-source library built on Numpy, SciPy, and matplotlib. It provides efficient tools for predictive data analysis supporting machine learning algorithms. In this thesis different internal cluster evaluation metrics are borrowed from Scikit-learn.

```
# Distance measure can either be 'Euclidean' or 'Levenshtein'
DISTANCE_MEASURE = 'Euclidean'

# Use the selected distance measure in its traditional or augmented form
AUGMENTED = True

# Set number of clusters
K = 4

# If set False, initial centroids will be selected randomly
ADVANCED_INITIALIZATION = False

# Discovery algorithm can be 'alpha', 'heuristic', or 'inductive'
MINER = 'heuristic'

# Filetyp can be 'xes' or 'csv'
FILETYPE = 'xes'

LOGPATH = "logs/variants/BPI_Challenge_2018_sampled_variants.xes"
AUGMENTED_LOGPATH = "logs/augmented/BPI_Challenge_2018_sampled_variants_augmented.csv"
```

Figure 4.1: Business object-centric trace clustering code usage example in Python.

For the calculation of the distance between two traces, we implemented the Euclidean distances as well as the Levenshtein distance. Both can either be computed using the traditional approach or using our novel, augmented approach.

For the calculation of the fitness, precision, generalization, and simplicity three different discovery algorithms from PM4PY can be used. The alpha miner is one of the most known process discovery algorithms. It finds a Petri net where all transitions are visible and unique. Additionally, all transitions correspond to classified events, like activities. It also finds an initial and final marking. The initial marking describes the status of the Petri net when an execution starts, and the final marking describes the same when an execution ends. The inductive miner is able to find a Petri net and a process tree. The basic idea is about detecting a 'cut' in the log and then recur on sub-logs. It makes use of hidden transitions, while each visible transition has a unique label. The heuristic miner algorithm acts on the Directly-Follows Graph to handle noise and find common constructs. The output is a Heuristics Net that contains the activities and the relationships between them. It can be converted to a Petri net [23].

Chapter 5

Evaluation

In traditional data clustering, evaluation is based on how well-resulted clusters fit the data. There are two ways to approach this problem: internal and external validation. External evaluation is possible if ground truth is available. It compares the clustering result with the ground truth clusters. The final output is regarded as 'good' if it is somehow similar to the ground truth. This evaluation technique is straightforward. However, a reference result is not always available.

Therefore, there needs to be another method, where the evaluation is purely based on the clustering result itself. This is much more realistic and efficient in many real-world scenarios and is called internal validation. It focuses on the structure of the resulting clusters using two main components: Compactness and Separation. The first focuses on intra-cluster similarity and measures how closely data points are grouped together within a cluster. The latter focuses on inter-cluster similarity and measures how different found clusters are from each other [8].

Additionally, process mining provides the opportunity to evaluate clusters in another way. Following the ideas of the control-flow perspective, process models are generated, which reflect the current observable process [6]. Since the goal of trace clustering is to provide more insightful and less complicated process models, the quality of the discovered process models should also be investigated. For this task, there are four dimensions to consider. Fitness indicates how much of the actual behavior is captured in the models. Precision addresses overly general models, while generalization addresses overly precise models. The last dimension, structure, assess the structure of the models [14].

5.1 Experimental Setup

In order to evaluate the business object-centric trace clustering approach, many experiment runs were conducted using eight different real-world event logs from different areas of application, including road traffic analysis, hospital billing processes, and declarations, which are available in the 4TU data repository. Table 5.1 provides an overview of the number of trace-variants and unique events, business objects, and actions of complete event logs while table 5.2 shows the same but for sampled event logs. Per event log, experiments were conducted eight times using random initialization and eight times using the advanced initialization for each k of value 3, 4, and 5. In order to reduce the size of the event logs and to minimize run time, the event logs were filtered based on trace variants. Particularly large ones were additionally sampled uniformly. Afterward, the resulting event logs were labeled using the semantic information extraction approach by Rebmann. All results were achieved using a Mac-Book Pro 2018 2,3 GHz Quad-Core Intel i5 processor and 8GB of RAM. VS Code was selected as the working environment using Python 3.9.5.

Name	Trace-variants	Events	Business Objects	Actions
Hospital Billing	1020	18	13	22
Road Traffic	231	11	8	8
Prepaid Travel Cost	202	29	7	8
Request For Payment	89	19	5	8
Domestic Declarations	99	17	5	10

Table 5.1: Overview of the real-life event logs used for the evaluation

Name	Trace-variants	Events	Business Objects	Actions
BPI Challenge 2018	475	38	19	32
BPI Challenge 2018	96	31	14	28
BPI Challenge 2019	1198	38	25	23
BPI Challenge 2019	160	33	21	22
Hospital Billing	204	17	11	17
International Declarations	251	34	9	11

Table 5.2: Overview of the sampled event logs used for the evaluation

As mentioned earlier, we used k-means clustering as our chosen clustering algorithm. To see the effects of exploiting semantic information for trace clustering, a baseline for our novel approaches is necessary. Therefore, we will be using the

Euclidean and Levenshtein distance in their traditional version on the original event logs as the baseline. The clustering results of our new approaches will be compared to their corresponding baseline considering internal validation as well as process model evaluation.

5.2 Measures

In order to evaluate the resulting clusters of a clustering algorithm using internal validation, there are multiple measures. While some evaluate either separation or compactness, there are also measure that reflect both indicators at the same time. Three of these measures are the Davies-Bouldin index, the Silhouette index, and the Calinski-Harabasz index. While we implemented all three, we focus our internal validation on only one score, the Silhouette index, due to its bounded score:

$$S = \frac{1}{NC} \sum_{i} (\frac{1}{n_i} \sum_{x \epsilon C_i} \frac{b(x) - a(x)}{max[b(x), a(x)]}) \tag{5.1}$$

where

$$a(x) = \frac{1}{n_i - 1} \sum_{y \epsilon C_i, x \neq y} d(x, y) \tag{5.2}$$

and

$$b(x) = min_{j \neq i} [\frac{1}{n_j} \sum_{y \epsilon C_j} d(x, y)]. \tag{5.3}$$

The Silhouette index uses the pairwise distance between all objects within a cluster for numerating compactness. With the average distance of objects to alternative clusters, the denominator measures separation. With +1 being the best and -1 the worst score, the Silhouette index indicates how highly dense and well separated the clusters are while scores around zero indicate overlapping clusters [8].

In order to evaluate the quality of the resulting process models, the different indicators need to be defined. For the calculation of the process models' fitness, the token-based variant is used, which is based on the new and faster version of token-based replay fitness. For the calculation of the process models' precision, the token-based variant is also used, which is faster, since it is based on heuristics.

5.3 Internal Validation

After conducting eight runs for each event log and setting, we calculated the average Silhouette scores for each event log and distance measure, as well as the overall

average, as displayed in table 5.3. Additionally, diagrams for each event log and number of clusters can be found in the appendix.

	Euclidean Distance		Levenshtein Distance	
Event Log	Traditional	Augmented	Traditional	Augmented
BPI Challenge 2018	0.3356	0.3231	0.1285	0.1051
BPI Challenge 2019	0.6576	0.5425	0.1736	0.1377
Hospital Billing	0.1727	0.2081	0.0955	0.0991
Road Traffic	0.3164	0.4016	0.1814	0.1990
Prepaid Travel Cost	0.1707	0.3214	0.1611	0.1936
Request For Payment	0.1918	0.2916	0.2275	0.2993
Domestic Declarations	0.1695	0.2866	0.2394	0.2981
International Declarations	0.1692	0.2932	0.1223	0.1699
Average	0.2729	0.3335	0.1662	0.1877
Difference	0.0606		0.0215	

Table 5.3: Overview of the average Silhouette scores of all distance measures and event logs

Looking at the Silhouette scores of the Euclidean and the augmented Euclidean distance, a general improvement of the cluster quality can be seen. In six of the eight event logs, there is a clear improvement for all k except two outliers, when using the smallest amount of clusters and the advanced initialization technique. One of the event logs does not show any major differences between both distance measures. Nevertheless, there is also one event log with better Silhouette scores using the traditional Euclidean distance. All in all, the augmented Euclidean distance improved the Silhouette scores on average by 0.0606.

When looking at the results of augmented Levenshtein distance and its baseline, the first impression is that in general, all scores are slightly lower and more similar. However, the augmented Levenshtein distance achieves higher scores in most cases which is also supported by the average score of all runs while it seems to struggle with the same two logs as the augmented Euclidean distance. All in all, the augmented Levenshtein distance improved Silhouette scores by 0.0215.

It seems that some event logs provide a great application of our novel approaches, while others do not. Therefore, we will take a deeper look into our experiments, in order to develop a better understanding of the influence of the number of business objects. Additionally, we are going to compare run times and see what consequences our novel approaches have on the run time of the clustering process. In order to do so, we are going to answer the following three questions:

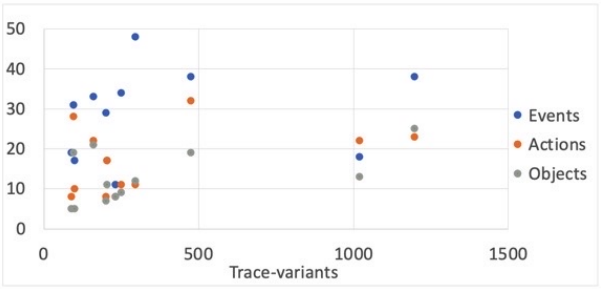

Figure 5.1: Relation between number of semantic roles, events, and trace-variants

- **Question 1:** How does the number of business objects influence the quality of the clusters?
- **Question 2:** How different are the run times? Is it dependent on the number of business objects?
- **Question 3:** Does a relatively long run time indicate a better quality of the clusters?

5.3.1 Question 1: How does the number of business objects influence the quality of the clusters?

As seen in table 5.1 and table 5.2, the evaluated event logs differ greatly in their number of unique events which also influences the number of possible business objects and actions. However, the number of semantic roles has no correlation to the number of trace-variants, as seen in figure 5.1. A perfect example of this fact is the event log BPI 2018 and BPI 2019. Even when greatly sampled, the number of semantic roles barely changes.

First, we look at the augmented Levenshtein distance. Figure 5.2 shows the relation between the number of business objects and Silhouette score improvements using the augmented Levenshtein distance. Each point refers to the result of one event log. The figure indicates a correlation between an increasing number of business objects and a decreasing Silhouette Index difference without any outliers. However, while the augmented approach achieves a cluster quality improvement when working with fewer business objects, it also provides barely any disadvantage when working with a high number of them.

When comparing the average distances of each event log with each other, we can see that the augmented Levenshtein distance decreases the distance between

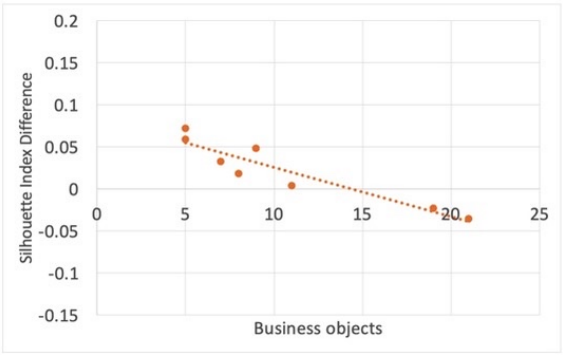

Figure 5.2: Relation between number of business objects and Silhouette score improvement using the augmented Levenshtein distance

traces by about 8%. This effect is a result of the fact that in the case of the same business object but different action the distance only increases by 0.5. In the case of the traditional Levenshtein distance, this would be represented by two different activities and therefore a distance increase of 1. With a rising number of business objects this effect, however, reduces resulting in very similar or identical average distances, which also explains the similar Silhouette scores with those event logs.

Second, we look at the augmented Euclidean distance. As seen in figure 5.3, there is also a negative correlation between an increasing number of business objects and the difference of the Silhouette index without a single outlier as all points are almost on the trend-line. The trend-line is also much steeper in comparison to figure 5.2.

A quick example might explain this phenomenon: trace1 **{A1}** and trace2 **{B1}**. Using the traditional Euclidean distance and our novel augmented Euclidean distance, we receive the following two distances.

- **Traditional:**

$$d_t = \sqrt{1+1} = \sqrt{2} \tag{5.4}$$

- **Augmented:**

$$d_a = \sqrt{1} + \sqrt{1} = 2 \tag{5.5}$$

In general, our approach computes larger distance, since we apply the Euclidean distance on smaller values and sum them afterward. Therefore, working with a large number of business objects results in greater distances even for similar traces. The average distance of our novel approach is larger for every event log.

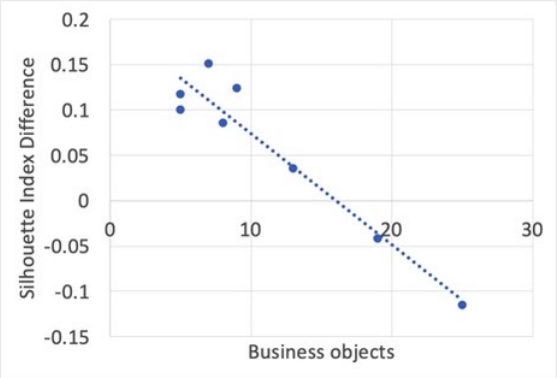

Figure 5.3: Relation between number of business objects and Silhouette score improvement using the augmented Euclidean distance

All in all, the relative distance to its baseline increases by about 70%, independent of the number of business objects. Resultantly, clusters will lose in compactness, since traces are by concept more distant to each other. However, when working with fewer business objects, this phenomenon seems to be overshadowed by the increasing separation of dissimilar traces.

5.3.2 Question 2: How much do the run-times differ? Is it dependent on the number of business objects?

In order to answer these questions, we set k to four and used random initialization. Into the run-time all steps of the four approaches were considered, including the loading of the event log, transforming traces into the necessary vectors or lists, selecting random initial centroids, all clustering iterations, and the calculation of the full distance matrix.

First, we look again at the augmented Levenshtein distance. Figure 5.4 shows the relative run-time increase of each event log, represented by the dots. As seen in this figure, the relative run time of the augmented Levenshtein distance has no correlation to the number of business objects. However, our novel approach results in a 5% increase in the run-time.

In contrast, the augmented Euclidean distance shows a clear increase of the relative run-time compared to the traditional measure, as seen in figure 5.5. While the run-time tends to stay around five to ten times the run-time of the baseline, the relative run-time reaches an increase of about 2000% if we go up to or even over 20 possible business objects.

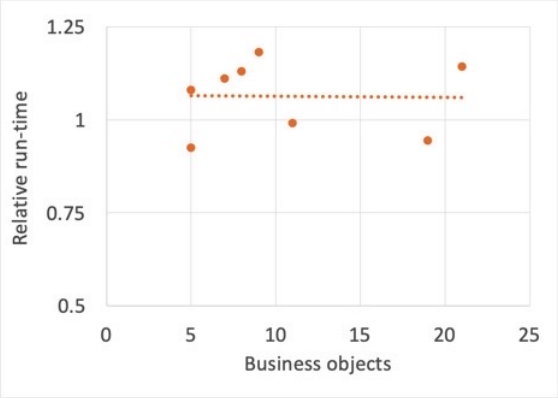

Figure 5.4: Relative run-time change of the augmented Levenshtein distance in comparison to its baseline

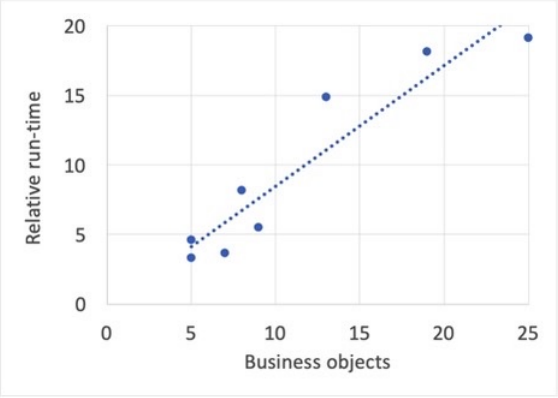

Figure 5.5: Relative run-time change of the augmented Euclidean distance in comparison to its baseline

5.3.3 Question 3: Does a relatively longer run-time indicate a better quality of the clusters?

Since the run-time of the augmented Levenshtein distance does not change relative to the number of business objects and is, in general, only about 5% longer than the regular Levenshtein distance, the question has no application considering this distance measure.

However, the effect of the augmented Euclidean distance is the complete opposite: The greater the relative run-time increase, the worse the cluster quality. As described in the two earlier questions, the improvement of the cluster quality keeps declining with a growing number of semantic roles. While the augmented Euclidean distance provides significantly better scores with ten or fewer business objects, there are no benefits as soon as we surpass 20 business objects. Additionally, the relative run-time increase of our approach also grows from five to almost 20 with an increasing number of business objects. Therefore, a high run-time does not only return results much slower but also provides worse results.

5.4 Process Model Evaluation

After successfully partitioning the event logs into three, four, or five clusters, process models are created for each of the clusters separately using two of the implemented discovery algorithms. Out of the eight runs for each event log and number of clusters k, we discovered all process models four times using the inductive miner algorithm and four times using the heuristic miner algorithm. For each of the four indicators, fitness, precision, generalization, and simplicity, we first computed the average of all clusters within one run and then the average of all four runs. Using these final scores, we present the difference between our novel approaches and their baselines.

5.4.1 Inductive Miner Discovery

Starting with the results of the augmented Levenshtein distance and its baseline using the inductive miner discovery algorithm in figure 5.7, we can see an overall improvement of the models' quality in the event logs where the cluster quality was also improved. The scores of the event logs with a high number of business objects overshadow the overall results. However, there are no major differences in the scores.

Second, we look at the results of the inductive miner discovery algorithm, as shown in figure 5.8. In general, the augmented Euclidean distance seems to improve the process model quality. Similar to the results of the Levenshtein distances,

LOG	BPI 2018	BPI 2019	Hospital Billing	Road Traffic	Prepaid Travel Cost	Request For Payment	Domestic Ded.	International Ded.	AVERAGE
Fitness	0.00021	-0.00014	0.00000	0.00000	0.00000	0.00000	-0.00020	0.00000	-0.00002
Precision	-0.00423	-0.01964	-0.00182	0.00684	0.00458	-0.02174	0.01636	0.00481	-0.00185
Generalization	0.00218	-0.00916	0.00296	-0.00154	0.00222	0.01058	-0.00108	0.00008	0.00078
Simplicity	-0.01424	-0.01247	0.00193	0.00235	-0.00138	0.00604	0.00258	0.00547	-0.00121
AVERAGE	-0.00402	-0.01035	0.00077	0.00191	0.00136	-0.00128	0.00441	0.00259	-0.00058

Figure 5.6: Average improvements of the process models' quality using the augmented Levenshtein distance and the inductive miner discovery algorithm

LOG	BPI 2018	BPI 2019	Hospital Billing	Road Traffic	Prepaid Travel Cost	Request For Payment	Domestic Ded.	International Ded.	AVERAGE
Fitness	-0.00006	-0.00099	0.00000	0.00000	0.00014	0.00000	-0.00022	0.00000	-0.00014
Precision	0.00031	-0.01555	0.02196	0.02385	0.02080	0.00405	0.02477	-0.00768	0.00906
Generalization	0.03666	0.00086	-0.01509	-0.00984	0.00453	-0.00306	0.00072	-0.00665	0.00102
Simplicity	0.02343	-0.00461	0.00403	0.00304	0.01226	-0.00464	-0.01793	-0.00751	0.00101
AVERAGE	0.01508	-0.00507	0.00273	0.00426	0.00943	-0.00091	0.00183	-0.00546	0.00274

Figure 5.7: Average improvements of the process models' quality using the augmented Euclidean distance and the inductive miner discovery algorithm

the fitness was always 1.0 or only barely lower, so there is no real difference visible. For all other indicators, the augmented Euclidean distance achieves on average slightly better scores. However, there is no correlation between a better process model quality and a better cluster quality.

5.4.2 Heuristic Miner Discovery

First, we look again at the result of the augmented Levenshtein distance and the heuristic miner discovery algorithm, as seen in figure 5.8. Even though there are many improved scores as well as slightly worse scores, the overall scores were all improved by the augmented Levenshtein distance. Nevertheless, there are no specific areas that show constant improvement.

Second, we look at the results of this miner algorithm using the augmented Euclidean distance, displayed in figure 5.9. Starting with the general impact on

LOG	BPI 2018	BPI 2019	Hospital Billing	Road Traffic	Prepaid Travel Cost	Request For Payment	Domestic Ded.	International Ded.	AVERAGE
Fitness	-0.05083	-0.00823	0.01922	-0.00306	0.01153	0.02961	0.02835	-0.00147	0.00314
Precision	0.01393	0.03379	0.05926	-0.03487	0.00290	-0.00973	-0.00100	-0.00890	0.00692
Generalization	0.00330	-0.00322	0.00642	0.00418	0.00049	-0.00243	0.00872	0.00959	0.00338
Simplicity	0.00951	0.00209	-0.00331	0.01381	0	0	0	-0.00206	0.00251
AVERAGE	-0.00602	0.00611	0.02040	-0.00498	0.00373	0.00436	0.00902	-0.00071	0.00399

Figure 5.8: Average improvements of the process models' quality using the augmented Levenshtein distance and the heuristic miner discovery algorithm

LOG	BPI 2018	BPI 2019	Hospital Billing	Road Traffic	Prepaid Travel Cost	Request For Payment	Domestic Decl.	International Decl.	AVERAGE
Fitness	-0.01526	-0.01767	-0.02011	0.02594	0.06351	0.06126	0.07089	0.07383	0.03030
Precision	0.04340	-0.04751	-0.04134	0.06757	0.11058	0.02277	0.00902	-0.02382	0.01758
Generalization	-0.00696	-0.02047	-0.00832	-0.00197	0.01433	-0.02900	0.00499	-0.02601	-0.00918
Simplicity	-0.01115	-0.02256	0.01067	-0.00253	0	0	0	0.00111	-0.00306
AVERAGE	0.00251	-0.02705	-0.01478	0.02225	0.04711	0.01376	0.02123	0.00628	0.00891

Figure 5.9: Average improvements of the process models' quality using the augmented Euclidean distance and the heuristic miner discovery algorithm

the process models' quality, it seems to increase the fitness and precision while decreasing the generalization and simplicity. However, the decreasing factors are smaller than the increasing ones. Therefore, the overall average shows a slight improvement.

Looking at specific event logs, we can see that the improvements are generally in line with the improvements of the Silhouette scores. Event logs with a higher cluster quality do also achieve higher scores, especially in fitness and precision, while the ones with a worse cluster quality also cannot improve the models' quality.

Chapter 6

Conclusion

With the emergence of objects-centric process mining, other areas in the domain of process mining need to adopt. Although research on process discovery algorithms using an object-centric approach achieve improving results, there is also a motivation to project this approach on trace clustering, since smaller event logs do not only reduce the amount of time needed to discover a single model but also improves the quality of that model. Therefore, we propose two business object-centric approaches to measure the distance between two traces. These approaches extend two prominent distant measures, the Levenshtein distance and the Euclidean distance, based on semantic roles and are designed to achieve a higher cluster quality.

The business object-centric approach is evaluated on eight real-life event logs, using k-means clustering based on the distances between traces computed using our novel measures the augmented Levenshtein distances and the augmented Euclidean distance. First, the resulting clusters were evaluated in terms of internal validation and run-time against their corresponding baseline. Second, using the heuristic miner and inductive miner discovery algorithms, the process models of all resulting clusters were evaluated on fitness, precision, generalization, and simplicity.

The results of the augmented Levenshtein distance showed that it provided the ability to achieve better quality clusters when working with few business objects without losing quality when working with a large number of business objects. Additionally, it barely reduces the performance by increasing the run-time by only about 5%.

The results of the augmented Euclidean distance showed an even greater improvement of the clusters' quality, providing better separation between clusters when working with a small number of business objects. However, with an increasing number of business objects, the improvement of the clusters' quality reduces,

while the relative run-time keeps increasing.

All in all, the limiting factor of the augmented Levenshtein and especially of the augmented Euclidean distance is the number of unique business objects in the event log. Event logs with a large number of this semantic role are not suitable since this results in too many 'business object vectors'. Consequently, the average distance increases too much in order to build dense clusters. Additionally, the run-time can increase by 2000% in comparison to the traditional Euclidean distance. However, new event logs that aim for object-centric process mining, provide the necessary feature of only a few business objects to achieve an improvement of clusters' and process models' quality.

The second part of the evaluation considered the quality of discovered process models using the inductive and heuristic miner algorithms. While the scores of the novel approaches were generally higher, the differences were not major and could not reflect the cluster quality improvements completely. However, since the process model discovery is not object-centric, it does not take advantage of the semantic roles, our trace clustering approaches are based on. The clustering might be better, however, the traces within a cluster are still very similar in a non-object-centric approach. Resultantly, the process models are rather identical. Therefore, the next step to even better process models would be taking our clustering results as an input for object-centric process model discovery approaches.

Chapter 7

Future Work

The are several possibilities for future work based on our novel approaches. First, the role of the clustering algorithm can be the subject of further studies. Therefore, the proposed distance measures can be used with different clustering algorithms, such as the K-nearest neighbor (KNN) or even hierarchical clustering techniques.

Second, to take an even deeper approach into object-centric process mining, the traces from the resulting clusters of our approach can be taken as input for discovering object-centric behavioral constraint models (OCBC) or object-centric Petri nets.

Third, as Rebmann and van der Aa described in their semantic information extraction approach, there are not only business objects and actions, but also additional semantic roles: Actors and passive resources. Resultantly, object-centric trace clustering approaches are not only limited to business objects and actions. With possibly adding actors or passive resources, how distances measures weight different semantic roles can be chosen completely freely, opening a wide range of possibilities. To provide a small glance into possible approaches, we extended our augmented Levenshtein distance and augmented Euclidean distance by including actors with the same weight as actions. In the case of the augmented Levenshtein distance, actors are considered similar to actions when the business object is identical. If both semantic roles are different, the substitution will be 1. If only one semantic role is different, the substitution cost will be 0.5. Only if both are identical, the substitution cost will be 0. In the case of different business objects, insertions, or deletions, the distance will be increased by 1. In the case of the augmented Euclidean distance, the adjustment is simpler. We do exactly the same, as we did with the actions, and simply add more dimensions to the business objects vectors. However, the possibilities to adjust these distance measures are endless.

Bibliography

[1] Ali Ridho Barakbah and Yasushi Kiyoki. A pillar algorithm for k-means optimization by distance maximization for initial centroid designation. In *2009 IEEE Symposium on Computational Intelligence and Data Mining*, pages 61–68, 2009.

[2] A. Berti, S.J. van Zelst, and W.M.P. van der Aalst. Process mining for python (pm4py): bridging the gap between process-and data science. *Proceedings of the ICPM Demo Track 2019, co-located with 1st International Conference on Process Mining (ICPM 2019)*, pages 13–16, 2019.

[3] Nicolas Bissantz and Jürgen Hagedorn. Data mining (datenmustererkennung). *WIRTSCHAFTSINFORMATIK*, 51:139–144, 2009.

[4] R. P. Jagadeesh Chandra Bose and Wil M. P. van der Aalst. Trace clustering based on conserved patterns: Towards achieving better process models. In Stefanie Rinderle-Ma, Shazia Sadiq, and Frank Leymann, editors, *Business Process Management Workshops*, pages 170–181, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[5] R. P. Jagadeesh Chandra Bose and Wil M.P. van der Aalst. *Context Aware Trace Clustering: Towards Improving Process Mining Results*, pages 401–412.

[6] Dusanka Dakic, Darko Stefanović, Ilija Cosic, Teodora Lolić, and Milovan Medojevic. *Business Process Mining Application: A Literature Review*, pages 0866–0875. 01 2018.

[7] Giuseppe Greco, Antonella Guzzo, Luigi Pontieri, and Domenico Saccà. Discovering expressive process models by clustering log traces. *Knowledge and Data Engineering, IEEE Transactions on*, 18:1010– 1027, 09 2006.

[8] M. Hassani and T. Seidl. Using internal evaluation measures to validate the quality of diverse stream clustering algorithms. *Vietnam Journal of Computer Science*, 4:171–183, 2017.

[9] Vladimir Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10:707–710, 1966.

[10] Guangming Li and Renata Medeiros de Carvalho. Process mining in social media: Applying object-centric behavioral constraint models. *IEEE Access*, 7:84360–84373, 2019.

[11] R. S. Mans, W.M.P. van der Aalst, and R. J. B. Vanwersch. Process mining in healthcare : opportunities beyond the ordinary. *BPM reports*, 1326, 2013.

[12] Arpita Nagpal, Arnan Jatain, and Deepti Gaur. Review based on data clustering algorithms. In *2013 IEEE Conference on Information Communication Technologies*, pages 298–303, 2013.

[13] Adrian Rebmann and Han van der Aa. Extracting semantic process information from the natural language in event logs. *CAiSE 2021: Advanced Information Systems Engineering*, pages 57–74, 2021.

[14] A. Rozinat, A. K. Alves de Medeiros, C. W. Günther, A.J.M.M. Weijters, and W.M.P. van der Aalst. Towards an evaluation framework for process mining algorithms. *BPM reports*, 123:142, 2007.

[15] M. Song, C.W. Günther, and W.M.P. van der Aalst. Trace clustering in process mining. *Business Process Management Workshop*, 362:109–120, 2009.

[16] T. Soni Madhulatha. An overview on clustering methods. *IOSR Journal of Engineering*, 2(4):719–725, 2012.

[17] W. M. P. van der Aalst. Object-centric process mining: Dealing with divergence and convergence in event data. *Software Engineering and Formal Methods.*, pages 3–25, 2019.

[18] W. M. P. van der Aalst and A. Berti. Discovering object- centric petri nets. *Fundamenta Informaticae*, 175:1–40, 2020.

[19] Wil Van Der Aalst. Process mining. *Communications of the ACM*, 55(8):76–83, 2012.

[20] W.M.P. van der Aalst. Process mining: overview and opportunities. *ACM Trans. Manage. Inf. Syst. 3, 2, Article 7*, page 17, 2012.

[21] W.M.P. van der Aalst and et al. Process mining manifesto. *BPM 2011 Workshops, Part I*, pages 169–194, 2012.

[22] R. Wagner and M. Fischer. The string-to-string correction problem. *J. ACM*, 21:168–173, 1974.

[23] A. Weijters, V. Aalst, and A. K. A. D. Medeiros. Process mining with the heuristicsminer algorithm. 2006.

[24] Lu Xixi, S.A. Tabatabaei, M. Hoogendoorn, and H.A. Reijers. Trace clustering on very large event data in healthcare using frequent sequence patterns. 2020.

Appendix A

Program Code / Resources

The source code, a documentation, some usage examples, and additional test results are available at GitHub: ”https://github.com/constantin1101/BusinessObjectCentricTraceClustering”.

They as well as a PDF version of this thesis is also contained on the USB attached to this thesis.

Appendix B

Further Experimental Results

In the following, there are the Silhouette scores of all event logs. The blue line represents the scores of the augmented Euclidean distance and the orange line represents the scores of the augmented Levenshtein distance, while the grey lines represent the corresponding baselines. The demonstrated scores for each number of clusters k is the average of 16 runs, including 8 runs using random initialization and 8 runs using the advanced initialization.

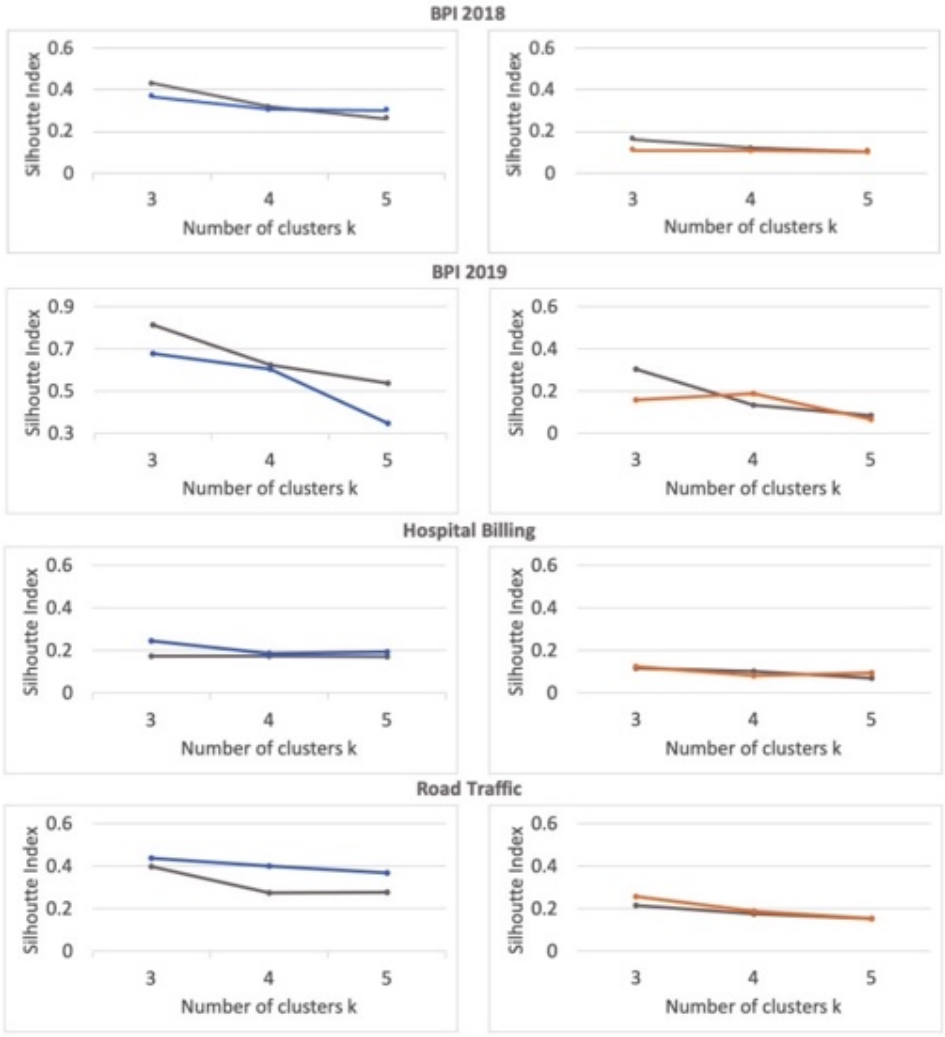

Figure B.1: Average Silhouette scores of the event logs BPI 2018, BPI 2019, Hospital Billing, and Road Traffic

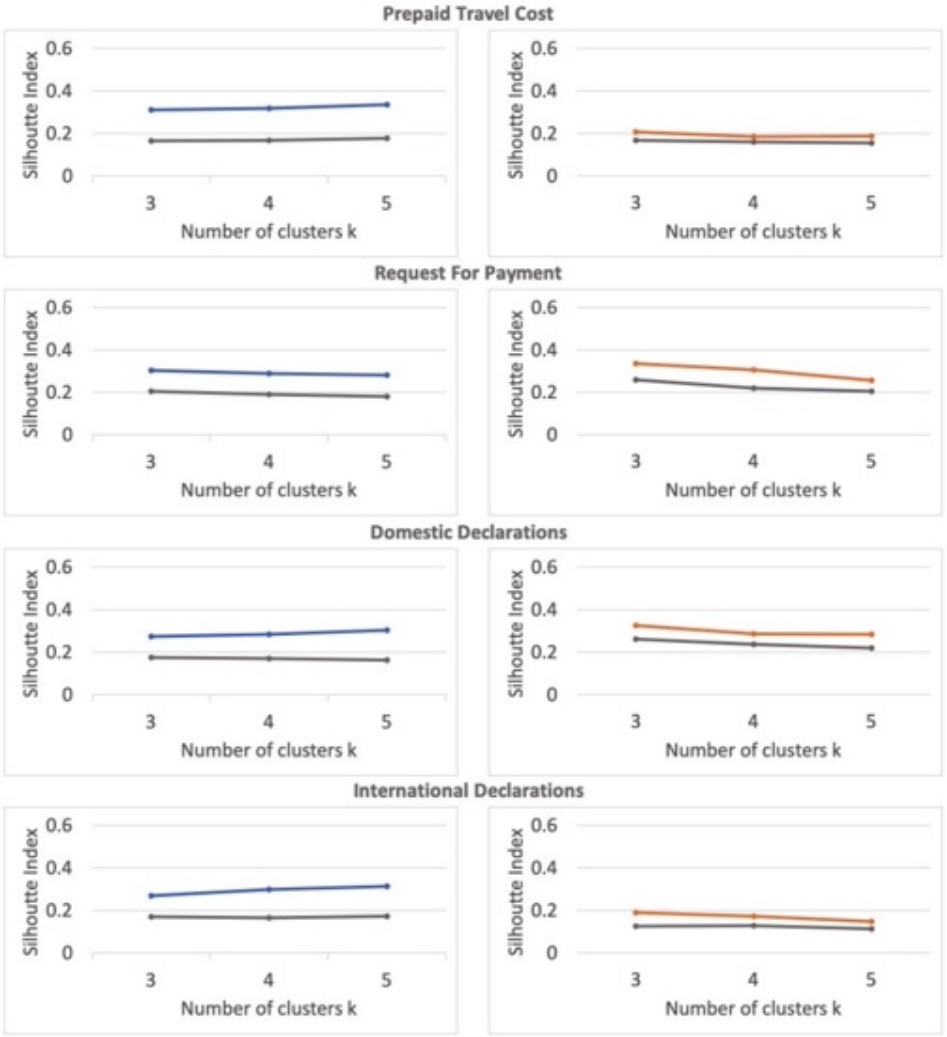

Figure B.2: Average Silhouette scores of the event logs Prepaid Travel Cost, Request For Payment, Domestic Declarations, and International Declarations.

Ehrenwörtliche Erklärung

Ich versichere, dass ich die beiliegende Bachelorarbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Mannheim, den 21.09.2021 Unterschrift