Linguistic summarization for process analysis
the development of a prototype

Munneke, J.F.C.

Award date:
2017

EINDHOVEN UNIVERSITY OF TECHNOLOGY & KPMG

Linguistic Summarization for process analysis

The development of a prototype

Master Thesis

Student:	**J.F.C. Munneke (0817562)**
Supervisors TU/e:	**A.M. Wilbik**
	D. Fahland
	R.M. Dijkman
Supervisor KPMG:	**E. Ramezani**

24-07-2017

Abstract

This master thesis investigates a new approach that can be used in analyzing general business processes. Current process discovery techniques mainly focus on the so-called control-flow perspective. Nevertheless, it might be interesting to also look at the other perspectives of the dataset, such as time and resources. The approach used in this master thesis is based on linguistic summarization, a technique used to generate statements in natural language, that describe characteristics of the dataset. Linguistic summarization has already been applied successfully on several datasets that consist of attribute-value pairs. However, the technique is only investigated sparsely on process data, which contains a time-line and causal relationships between activities. In addition, this thesis focuses on the generation of sentences that focus on a particular sequence within a process, instead of the complete case, which can provide more detailed insights into specific parts of the process. Event logs can contain a lot of data and, therefore, the created algorithm must deal with the data in an efficient manner. The technique is evaluated using two case studies. The first case study is performed on thirteen participants, where first a demonstration is given, after which the participant had to fill in a questionnaire. The other case study is performed for audit purposes, where the algorithm was evaluated with a process expert. The evaluation of the algorithm shows that most participants see the usefulness of linguistic summarization, and they intend to use it.

Preface

This master thesis is performed as final requirement for the master 'Business Information Systems', that I have been studying for the last two years at the Eindhoven University of Technology (TU/e). This master thesis has been conducted for the past six months, and is performed in collaboration with KPMG, which gave me a wonderful time. My colleagues within KPMG really accepted me as an co-worker, instead of just an intern, and were really eager to help me with any kind of problems. Besides the many games of table football, I joined many delightful events, such as the yearly ski-trip. Because of this, I also had time to relax, which ensured a reduced stress level. Therefore, I was highly motivated to continue with the thesis. In this section, I would like to take the opportunity to thank everyone that helped me in finalizing the master thesis; for all the valuable insights, feedback, and motivation.

First of all, I want to thank Anna Wilbik, who acted as my first supervisor within the TU/e. She was already familiar with the topic and was, therefore, able to help me understand the topic and provide a lot of valuable insights during the bi-weekly meetings. In addition, I want to thank Dirk Fahland, who acted as my second supervisor within the TU/e, for both the feedback for the final draft, and the help during the project, if I had any problems. Thanks to Remco Dijkman, my third supervisor within the TU/e, for the introduction to the master thesis, the feedback for the final draft, and the help with the academic paper. Next, I would like to thank Elham Ramezani, who acted as my daily supervisor within KPMG. She helped me making this thesis available within KPMG, and had a lot of valuable insights during the process. Next to the supervisors, I want to thank Bert Scherrenburg and Manish Batra. Bert acted as a process expert for this thesis and introduced me to the case study, and Manish helped me developing the event log that was needed for the case study. Thanks to Erwin Raetsen for the valuable feedback and discussions during the process. I would also like to thank all participants that helped me with the evaluation of this master thesis.

Next, I want to thank my family for all the support, and for keeping me motivated. Especially thanks to my dad, for the feedback on the thesis. I want to thank my friends for the fun besides the projects, such that I was not only focused on my study, but made my time as student much more fun. Last, but definitely not least, I want to thank my girlfriend, Leanne, for all the moral support during my complete study, and for all activities we did, such that I was able to relax after a long week of work.

24-07-2017

Johan Munneke

Contents

List of figures

List of tables

1 Introduction

The amount of data that can be used to analyze business processes is increasing [1]. Therefore, getting and interpreting the results of common process mining techniques can be quite hard and time consuming. In practice, many processes have a high number of distinct activities, many data attributes, or a complex structure. Consequently, the automated discovery of such processes will possibly lead to so called 'spaghetti' models that are hard to comprehend in the first glance and require extensive post processing [1]. Another challenge in process discovery is dealing with different process perspectives. Currently, the dominant perspective in most process discovery techniques is the so-called control-flow perspective [2]. Nevertheless, it may be interesting to link this perspective to other perspectives of a process, such as time and resources. It may, for example, be interesting to find a correlation between the throughput time and the resources that are associated to a process. There exist already some process mining techniques that take the different perspectives into account, e.g. related with conformance checking [3], decision mining [4] [5], event correlation [6], and risk management [7]. However, none of these research papers focus on process discovery, where the output is obtained in natural language, while this research focuses mainly on this aspect.

This research proposes another way of gaining insight into business processes without the extensive post-processing that some process mining techniques require. It allows an easier interpretation of the results that process mining techniques provide, i.e. even for practitioners that are not familiar with process mining techniques. This research is based on linguistic summarization, a technique used to gain insight into a collection of data, by automatically generating statements in natural language, that describe characteristics of the dataset. Example sentences that can be generated using linguistic summarization on process data are *"In almost all cases, where Person A performed task X, the costs were high"* or *"In almost all cases, that contain a sequence like <ABC>, the costs were high"*. Such sentences can be used as an indication of root causes of high costs.

In recent years, linguistic summarization has been applied successfully in different areas, e.g. databases [8] [9], sensor data [10] [11] [12], texts [13], time series [14] [15] [16] [17], video fall detection [18] [19] [20] and web logs [21]. However, the datasets used in these research papers consist only of attribute-value pairs, which is different than process data, because process data has a time-line and causal relationships between activities [22]. The topic of linguistic summarization on process data is already introduced in [22], [23] and [24]. [22] introduces a research agenda of tasks that have to be taken into account when applying linguistic summarization on process data, [23] introduces several types of sentences that could be created for process data, and [24] focuses on sequences of actions and applies the technique on one single event log. However, the subject is only investigated sparsely and no generic algorithms exist yet that apply linguistic summarization on processes. As introduced in [22], there are many challenges that need to be addressed, such as: the structure of event logs is inconsistent; a clear and complete overview has to be generated about the process; and an event log can consists of a lot of sequences of events, which can increase the running time of an algorithm. This research is performed to address these challenges. In addition, the generation of sentences that focus on sequences, instead of the complete case, is only investigated sparsely in previous literature. This thesis focuses how sequence can be handled, such that sentences like "*In most cases, there was a large throughput time for a sequence like <ABC>*" can be investigated.

A technique is developed that uses linguistic summarization to generate statements about process data, that the user can use to gain insight into the process. These statements can be used by anyone that wants to analyze a business process, and might give interesting insights into the business process, that are hard, if not undoable, to find using other tools. The technique has been implemented in Python [25] and is applied on various datasets.

As part of the evaluation of this master thesis, two case studies are performed: one case study is related to auditing and the other case study is related to an appeal process of a Dutch municipality. The focus of these evaluations is on user acceptance, to determine whether the technique, and the resulting sentences, might be useful in the analysis of a business process.

1.1 Problem statement

This research focuses on getting insight into business processes by using linguistic summarization techniques. Since the research focused on business processes in general and is not context specific, the main research question is stated as follows:

How can linguistic summarization ease the analysis of (complex) processes?

Imagine a process with many distinct activities, cases or attributes. The automated discovery of such processes can lead to so called 'spaghetti' models that are hard to comprehend in the first glance and require extensive post processing [1]. The main research question focuses mainly on providing insights in an easy and intuitive way that would be difficult to obtain by merely using process discovery instead. Although the main contribution of the proposed technique will be for complex business processes, the simpler processes can also be analyzed using this technique.

To answer this research question, it is divided into additional sub-questions that all focus on different parts of the project. These sub-questions are described below.

1.1.1 Different type of protoforms

Linguistic summarization returns template based sentences. These template based sentences are also called protoforms [23]. The sentence *"In most cases, there was a large throughput time"* and *"In most cases, where there was a large throughput time, the costs are high"* have got a different structure and, therefore, belong to different protoforms. The first sentence creates a statement about all cases, where the second one is only focused on cases that have got a *large throughput time*. By creating protoforms, the sentences are written in a consistent manner, which is clearer for the user, and, in addition, statistics can be calculated in the same way for every protoform. Already many different protoforms have been investigated in prior literature. However, new protoforms may be required for the summarization of process data. The first sub-question focused on the investigation of different protoforms that one can have in process data. The sub-question is stated as follows:

1. *What protoforms can be useful to summarize process data?*

1.1.2 Data preprocessing

The technique proposed in this research is not context specific and, therefore, several preprocessing steps need to be performed on an event log. This section focuses on the preprocessing of the data and the sub-questions related to this part.

An event log contains information about events that occurred for a specific process. Every event is linked to a process instance, which is referred to as a case [1]. Cases can be distinguished using a case identifier. As discussed in the introduction, the structure of event logs is inconsistent. In one event log, the identifier of a case might be marked as 'Identifier', in another event log as 'Case ID', and in yet another event log as 'Column X'. When evaluating the event log, one has to make sure that the right columns are evaluated. Next to the different naming of columns, cells may be left blank, or the event logs may contain NULL values. Some preprocessing has to be done before certain features, such as the throughput time, can be calculated.

There are many features that can be relevant for business processes. Examples are throughput time, resources involved in the process, and paths taken. Some research must be performed to determine which features are relevant for process data. The sub-question related to this part of the process is stated as follows:

2. a. What features are relevant to analyze cases and sequences?

Features that may be relevant for one process can be irrelevant for other processes. Since this research is not meant for one single process, the selected features must be important for processes in general. The set of features have a direct influence on the results, and must therefore be as complete as possible. However, different features can be selected for different processes, for example, when events do not have an end time, the waiting time cannot be calculated.

To be able to create sentences like *"In most cases, there was a large throughput time"* or *"In almost all cases, when there was a short waiting time, there was Person A involved"*, a dictionary has to be created in which the meaning of the possible quantifiers (e.g. *most* and *almost all*) and features (e.g. *large throughput time* and *short waiting time*) is defined. The meaning of such quantifiers and features is stored in so-called linguistic labels, e.g. a throughput time of two days is considered as large for that process. Note that no linguistic labels have to be created for the involvement of *Person A*, since this is either true or false. To determine how to create linguistic labels, another sub-question is defined:

2. b. How to create linguistic labels for the selected features?

The labeling of features can be done manually, automatically or a combination of both. In the first case, the user has to define all labels, which can take quite some time and effort. In the second case, the labels are generated by use of an algorithm. In the last case labels are proposed by use of an algorithm, yet the user can change them. For both, the second and last case, an algorithm is needed. However, the algorithm can be less complex in the last case, since the user can still modify the linguistic labels.

1.1.3 Modelling

Next to the preparation of data, an algorithm is needed to generate sentences about an event log. These sentences are also called linguistic summaries. Event logs can be quite large and can contain many different cases and sequences. In addition, many features can be analyzed for every event log. The algorithm has to deal with this in an effective and efficient way. The sub-question related to this is formulated as follows:

3. *How to efficiently generate linguistic summaries of processes?*

The created summary has to be as complete and clear as possible. However, several challenges are involved in this part, for instance, an event log can contain many different sequences. One can look at a particular sequence, instead of looking at the complete case to get different insights into the process and find a correlation between attributes of a sequence and attributes of a case, e.g. *"In most cases, where the sequence <ABC> was performed by Person A, there was a large throughput time"*. Since there are probably many different sequences, both the running time and the results will be gigantic when looking at all possible sequences. Therefore, an intelligent choice has to be made on which sequences to focus.

Next to the selection of sequences, one has to deal with the data in an efficient way. A possible way to calculate features is to loop over the event log over and over, which is not very efficient. The algorithm has to deal with the data in an intelligent way, for example, by storing frequently used variables, using packages in a clever way, structuring the data, and pruning results.

1.1.4 Case study

As a final part of the project, the linguistic summaries have to be evaluated to determine their relevance. To be able to evaluate the results, two case studies are performed. The first case study is related to auditing. However due to privacy considerations, another case study is performed on the appeal process of a Dutch municipality, which is analyzed in detail in this thesis. More information about this event log is given in Section 1.4. The auditing related case study is performed to determine the usefulness of the linguistic summaries in a real business scenario, where the focus of this thesis is on user acceptance. The sub-question related to the evaluation is:

4. *How can the auditor make use of the linguistic summaries?*

The focus of the case study is to give insight into the use of linguistic summarization in analyzing a business process. The case study does not focus on the actual results provided by the algorithm, but on the manner the results are generated and whether these results could be useful for an auditor.

1.2 Scope

This thesis investigates the possibility of linguistic summarization and focuses on the creation of a first prototype of the algorithm. Sometimes choices had to be made, where multiple options were considered as valid. However, due to timing reasons, not all options could be investigated. Therefore, not all choices that have been made in this thesis are definitely the only correct option. Choices that had to be made were the kind of sentences that one could investigate, explained in more detail in

Chapter 3; features that are seen as important for an event log, explained in more detail in Chapter 5; and choices related to the implementation, e.g. how to define the membership functions, how to group sequences, and how to use the resulting groups of clusters, explained in more detail in Chapter 6 and 8.

1.3 Methodology

In this section, the project methodology, that is used to answer the research question and sub-questions, is explained.

The methodology is based on the Design Science Methodology [26], and is used for the creation of new and innovative artifacts, by iteratively add functionalities and test it against the requirements set. For every iteration, the tool can be extended for either rigor (using literature) or relevance (using practitioners). Using both literature and practitioners, the gap between rigor and relevance can be minimized, and a tool is developed that can be used by both parties [27]. Six phases are defined for this research, visualized in Figure 1. In addition, the deliverables of each phase can be found underneath the corresponding phase.

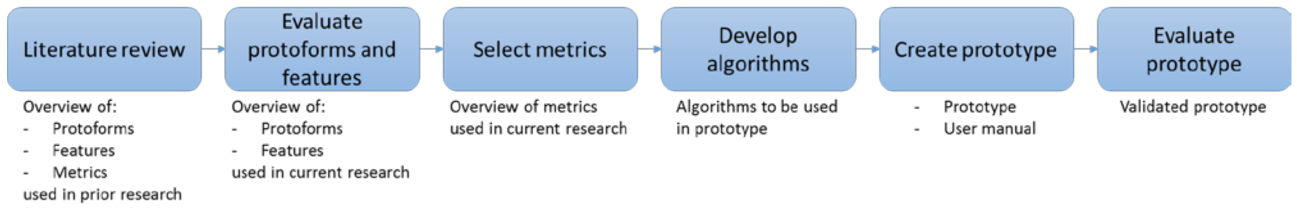

Literature review
Evaluate protoforms and features
Select metrics
Develop algorithms
Create prototype
Evaluate prototype
Overview of: - Protoforms - Features - Metrics used in prior research
Overview of: - Protoforms - Features used in current research
Overview of metrics used in current research
Algorithms to be used in prototype
- Prototype - User manual
Validated prototype

Figure 1: Project methodology

First, a literature review is performed to determine which protoforms, features, and metrics are found in prior research on linguistic summarization. The results are used as a starting point for the first three sub-questions, since it is determined what techniques show to be effective.

After the literature review is conducted, the protoforms and features, used in prior research, are evaluated on their applicability and usability for the current research. New protoforms and features are constructed, that are (partly) based on the constructed lists. the first sub-question (*What protoforms can be useful to summarize process data?*) and the first part of the second sub-question (*What features are relevant to analyze cases and sequences?)* are answered in this phase.

Next, the metrics are selected that are used for linguistic summarization on process data. The second part of the second sub-question (*How to create linguistic labels for the selected features?)* and the third sub-question (*How to efficiently generate linguistic summaries of processes?)* are answered during this phase.

After the metrics are selected, algorithms are developed, to be used in the prototype to be created. Algorithms focus on how to group sequences, how to prune the results, and how to generate sentences for the protoforms chosen.

During the ‘Create prototype’ phase, a prototype of a tool is constructed, that is able to generate linguistic summaries, focusing on an event log. An agile way of working is chosen for this phase of the project. First the least viable product is built, that can already be evaluated on different datasets. Depending on the evaluation, the algorithm is updated. For this project, the deployment is limited to the creation of a user interface in combination with a brief user manual. The user manual is provided in Appendix A.

After the prototype is built, it is evaluated and checked against all requirements. Two case studies are performed during this phase, and the final sub-question (*How can the auditor make use of the linguistic summaries?)* is answered. For the case study related to auditing, the technique is validated by a process expert, to determine the usefulness of the linguistic summaries for an auditor. For the case study related to the appeal dataset, it is investigated whether participants see the benefits of using linguistic summarization and whether the technique is usable.

1.4 Appeal dataset

As part of this thesis, a case study is performed on the appeal dataset of a Dutch municipality, which is based on [24]. When a citizen appeal for a decision that was made by the government, the appeal process is executed. Examples of such decisions are an application for a certain permit (like a building permit, demolition permit or tree cutting permit) or about compensation for traveling by school bus. Figure 2 shows the corresponding process model. This model is filtered for simplicity and only shows the most common flows that are taken. There are several variations on the flow, where for example only a part of the flow is executed. This model is created according to the Business Process Model and Notation (BPMN) semantics [28].

It can be seen, in Figure 2, that the process starts by registering the appeal. After the registration, a decision is made to either sent on the appeal when it cannot be handled or to continue with the main flow. If one chooses to continue with the main flow (confirms the reception), another decision has to be taken: The appeal is rejected, the citizen is asked to revise the appeal, or the documents are registered. When the citizen is asked to revise the appeal, this is archived and the process terminates. When the appeal is rejected, a draft advise can be written or the documents can be registered and it is processed the same as an approved appeal. In the last case, when the documents are registered, the appeal is discussed in a hearing, which can result in a withdrawal of the appeal or that advise is sent to the mayor and the alderman. When advice is sent, the mayor and the alderman make the actual decision which is documented in a dossier. In both cases the decision is archived and the process terminates.

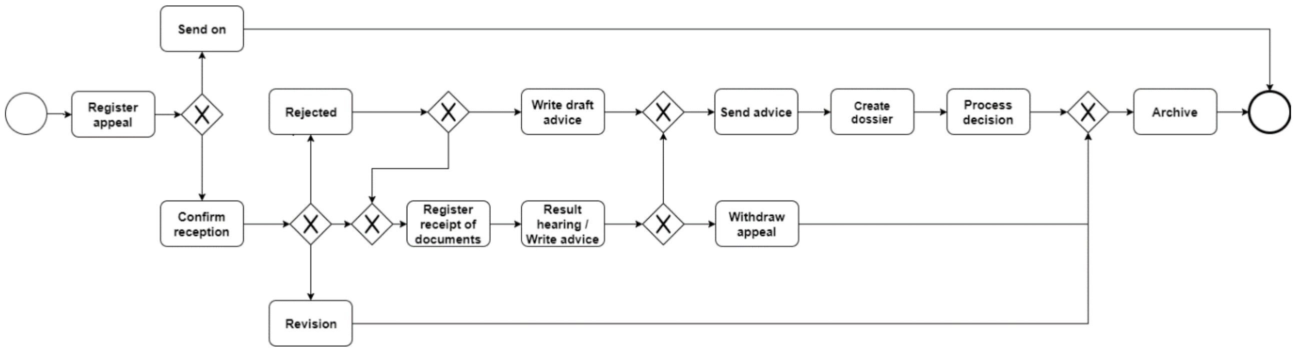


Figure 2 Appeal process

1.5 Report structure

This master thesis is structured using seven phases, as is shown in Figure 3. The thesis is structured such that most phases are handled within one chapter. However, since the implementation part is discussed more extensively, this phase is spread over three chapters.

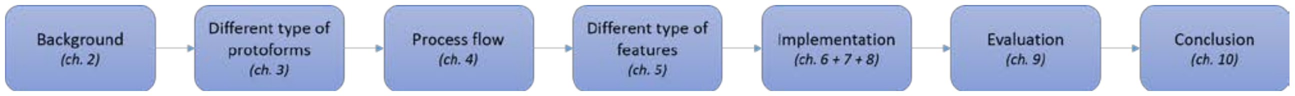

Background (ch. 2)
Different type of protoforms (ch. 3)
Process flow (ch. 4)
Different type of features (ch. 5)
Implementation (ch. 6 + 7 + 8)
Evaluation (ch. 9)
Conclusion (ch. 10)

Figure 3: Thesis outline

In Chapter 2, the background phase, information is provided that is needed to understand the remaining part of the thesis. Next, in Chapter 3, it is investigated what kind of protoforms might be interesting for business processes. Chapter 4 provides an overview of the process, that has to be executed to generate linguistic summaries, and serves as an outline for Chapter 5 till 8. The different types of features, that might be relevant to look at, are discussed in Chapter 5. The implementation phase consists of three parts: the implementation issues for the different protoforms are discussed in Chapter 6 and 7, and Chapter 8 discusses how the linguistic summaries are created. The evaluation of the technique is described in Chapter 9. Finally, concluding remarks about the master thesis are given in Chapter 10.

2 Background

This chapter focuses on background information that is needed to understand the remaining part of this thesis. Section 2.1 provides general information about linguistic summarization. Next, Section 2.2 shows the benefits of using fuzzy sets, and Section 2.3 introduces the meaning of protoforms, that can be used to create sentences.

2.1 Linguistic summarization

A linguistic summary is a sentence that is based on a template. These templates are referred to as protoforms [23]. As discussed in, for instance [22], [23], and [24], the principle of linguistic summarization has already been researched by many researchers. In this research, the method proposed by Yager [29] is used, that seems to be mostly explored in the literature.

Yager [29] defines that a protoform consist of three main aspects, namely: a summarizer, a quantifier and a measure of validity of truth.

- A summarizer is used to indicate the object of interest and can be about any feature that is available in the data, e.g. the size of a ball, the gender of a person, or the throughput time of a case. In this thesis *P* is used as an indicator of the summarizer;
- The quantifier is used to express the proportion of data which fulfil the summarizer. In this research, only relative monotonically non-decreasing quantifiers [30], such as many, most, or almost all are used. In this thesis *Q* is used as an indicator of the quantifier;
- The measure of validity of truth is used to express the validity of a sentence. In this thesis, this concept is called the truth value. Section 2.3 explains the truth value in more detail.

By combining these aspects, sentences like *“Most cases are long”* can be created. In this sentence *“Most”* is the quantifier (Q) and *“long”* is the summarizer (P). Note that the summarizer can be used to denote multiple features to generate sentences like *“Most cases are long and expensive”*.

Sentences like the ones shown above can be used to get a high level overview of the data. However, one can gain more detailed information by extending the sentences with a qualifier. [23] shows that a qualifier can be useful in the summarization of data.

- The qualifier can be used to define the scope of the sentence. The qualifier can focus on any feature available in the dataset, similar to the summarizer. When setting a feature as a qualifier, more detailed information about this feature can be gained. In this thesis, *R* is used as an indicator of the qualifier.

By adding the qualifier, sentences like *“Most expensive cases are long”* can be created, where *“expensive”* is the qualifier. Again, the qualifier can be about multiple features, to set a more specific scope, e.g. *“Most expensive and long cases are performed by Person A”*. In this sentence *“Most”* is the quantifier, *“expensive and long”* is the qualifier and *“performed by Person A”* is the summarizer.

2.2 Fuzzy sets

The example sentences shown in Section 2.1 are not very specific. For example, for the sentence *“Most cases are long”*, one might understand what is meant by this sentence, but not the exact meaning. The sentence *“80% of the cases took 20 days”* gives more detailed information. However, when one does not know any statistics about the duration of a case, the second sentence might be useless, i.e. it is not known whether 20 days is long or short. Next to this, when continuous values are used, like the duration of a case, all durations might slightly differ. Therefore, it might be more useful to set some ranges when the duration is for example short, average or long.

Suppose a user wants to see whether the duration of a case is short, average or long. For some durations, it might be unclear to which set the value belongs. If one uses crisp sets, the value must be either a member of the set or not, i.e. the duration is either short or not. However, by using fuzzy sets, a value has got a certain degree of membership to every set. The value might, for example, have a membership of 0.7 to the short set and a membership of 0.3 to the average set, which implies that the value belongs for 70% to the short set and for 30% to the average set. These membership values can be used in calculations, such as truth value. The definition of a fuzzy set is given below.

Definition Fuzzy set: “A fuzzy set is a class of objects with a continuum of grades of membership. Such a set is characterized by a membership (characteristic) function which assigns to each object a grade of membership ranging between zero and one.” [31]

2.3 Protoforms

In Section 2.1, two different kind of sentences are discussed. These sentences comply to the simple and extended protoform introduced in [23]:

- The simple protoform can be used for sentences like *“Most cases are long”* and is expressed as:

  $$Q\ y's\ are\ P \tag{1}$$

  where Q is the quantifier(e.g. most), y is the object the sentence is about (e.g. cases), and P is the summarizer (e.g. long).

- The extended protoform can be used for sentences like *“Most expensive cases are long”* and is expressed as:

  $$Q\ R\ y's\ are\ P \tag{2}$$

  where Q is the quantifier (e.g. most), R the qualifier (e.g. expensive), y is the object the sentence is about (e.g. cases), and P is the summarizer (e.g. long).

Sentences are protoform-equivalent if they have identical protoforms [32], e.g. the sentences *“Most cases are long”* and *“Most cases are expensive”* are built of the same protoform and are therefore protoform-equivalent. Protoforms are not only used to keep the created sentences consistent, but also to be able to make calculations more general. For example the measurement of truth, which expresses the validity of a sentence, can be measured in the same way for all sentences that are constructed of the same protoform.

There are many different methods to calculate the truth value, that all have advantages and disadvantages. For a comparison of different methods, see for example [33], [34], [35] or [36]. For this research, the initial method is chosen, which is based on Zadeh's calculus of quantified propositions [37], a method used to calculate the truth value of statements with quantifiers. Hence, the truth value of the simple protoform (1) is calculated as:

$$T(Q\ y's\ are\ P\ ) = \mu_Q\left(\frac{1}{n}\sum_{i=1}^{n}\mu_P(y_i)\right) \tag{3}$$

The truth value of the extended protoform (2) can be calculated as:

$$T(Q\ R\ y's\ are\ P\ ) = \mu_Q\left(\frac{\sum_{i=1}^{n}\mu_P(y_i)^\wedge\mu_R(y_i)}{\sum_{i=1}^{n}\mu_R(y_i)}\right) \tag{4}$$

Where μ_Q is the membership of the quantifier, μ_R the membership of the qualifier, μ_P the membership of the summarizer, n the number of objects (i.e. cases) in the data and ^ is the minimum operator. The definition of the membership functions are described in more detail in Section 6.2.

The truth values of both the simple protoform (3) and the extended one (4) can be used in the creation of linguistic summaries. One can for example indicate for what truth value the sentence is considered to be valid. In protoform (2) a qualifier is used to specify a specific part of the process. Not all cases satisfy the condition set in the qualifier, e.g. the sentence "Most expensive cases are long" only focuses on cases that are expensive. The proportion of objects satisfying qualifier R, called the degree of focus [14], can be calculated as:

$$d_f(Q\ R\ y's\ are\ P) = \frac{1}{n}\sum_{i=1}^{n}\mu_R(y_i) \tag{5}$$

By setting a minimal threshold for the degree of focus, sentences should at least concern a certain part of the data, i.e. when the degree of focus is set to 0.3, the proportion of objects satisfying the qualifier must be at least 30%. Sentences that are only about a few cases may not be very relevant and can give a wrong impression about the data. For example, the sentence *"Almost all expensive cases are long"*, where only one case is expensive. This sentence can be considered as a unique case and may not be wanted in the resulting summaries.

3 Possible protoforms

In Section 2.3, the simple (1) and extended (2) protoforms are introduced, together with formulas about their truth values, (3) and (4) respectively, and the degree of focus (5). In this chapter, these protoforms are linked to processes and additional protoforms are explained, that are based on prior research on linguistic summarization. First, it is determined what kind of statements can be relevant for process data, after which these statements are converted into protoforms. Appendix B provides an overview of which protoform is based on which article or other protoform. The statistics about all protoforms, in the scope of this thesis, can be found in Appendix C. All statistics are based on metric (3), (4) and (5). In Section 3.1 - 3.4, the different protoforms that may be relevant for processes are discussed. In Section 3.5 the protoforms that are used in this thesis are summarized.

3.1 Case focused protoforms

This thesis distinguishes two different forms of case focused protoforms: case focused protoforms for features related to the complete case and case focused protoforms related to sequences.

3.1.1 Case focused protoforms for features related to the complete case

The simple, case focused protoform [23] is a direct transformation of protoform (1) and is written as:

$$In\ Q\ cases, there\ was\ P \qquad (6)$$

For this protoform, sentences like *"In almost all cases, there was a short throughput time"* can be created. In this sentence, P is a specific attribute. However, P can also be a set of attributes, i.e. *"In most cases, there was a short throughput time, and Person A performed the activity Register appeal"*.

One may be interested in focusing on one (or multiple) specific feature(s). To be able to set certain conditions in a sentence, protoform (2) is transformed to the extended, case focused protoform [23] and is expressed as:

$$In\ Q\ cases, when\ condition\ R\ was\ fulfilled, there\ was\ P \qquad (7)$$

By using this protoform, sentences like *"In most cases, when Person A was involved, there was a large throughput time"* can be created, where more information is provided under which conditions there was a *large throughput time*. In this protoform, both R and P represent a set of attributes.

3.1.2 Case focused protoforms related to sequences

Since the process can be executed slightly different every time, the event log can contain many different sequences, i.e. different decisions are taken or events are executed in a different order. One may be interested in sequences that are performed by creating sentences like *"Most cases contain the sequence <Create dossier, Process decision>"*. When focusing on all sequences of the event log, both the running time of the algorithm and the amount of sentences can be gigantic [24]. To be able to deal with this, similar sequences are grouped together, which is explained in more detail in Chapter 7. When sequences are taken together, one can create sentences about the complete group instead of on specific sequences. The simple, case focused protoform, related to the containment of sequences [24], is written as:

$$Q\ cases\ contain\ a\ sequence\ like\ ABC \tag{8}$$

Note that the word *"like"* indicates that the sentence refers to all sequences of that particular group. This protoform can be used to get a high level overview of the sequences that are performed in the process. An example sentence of this protoform can be *"Most cases contain a sequence like <Send advice, Create dossier, Process decision>"*. In this case, the group <Send advice, Create dossier, Process decision> is created that consists for example of the sequences <Send advice, Create dossier, Process decision>, <Send advice, Create dossier>, <Create dossier, Process decision> and <Send advice, Create dossier, Process decision, Archive>.

Also for this protoform, one may be interested in setting certain conditions, like in protoform (7). The extended, case focused protoform, related to the containment of sequences [24], is written as:

$$Q\ cases, when\ condition\ R\ was\ fulfilled, contain\ a\ sequence\ like\ ABC \tag{9}$$

When protoform (9) is used, one can see what the influence of other features was on the sequences that were performed, e.g. when Person A was involved. However, the containment of a sequence can also be set as a condition. By doing so, two more protoforms are created:

$$In\ Q\ cases, that\ contain\ a\ sequence\ like\ ABC, there\ was\ P \tag{10}$$
$$Q\ cases, that\ contain\ a\ sequence\ like\ ABC, contain\ a\ sequence\ like\ XYZ \tag{11}$$

When protoform (10) is used, the influence of certain sequences can be seen on other features, e.g. "*In most cases, that contain a sequence like* <Send advice, Create dossier, Process decision>, there is a long throughput time", indicates that the throughput time of the complete case is long, if this sequence is contained. To see whether there is some correlation between certain sequences that were performed, protoform (11) can be used.

Note that protoforms (8), (9), (10) and (11) are all based on protoforms (6) and (7), where P and/or R is replaced by *"a sequence like ABC/XYZ"*.

3.2 Sequence focused protoforms

Instead of looking at the complete case, one can also focus on a specific part of the case. This distinction is made in [23]. The most simple form of the sequence focused protoforms is expressed as:

$$In\ Q\ cases, there\ was\ P_S\ for\ a\ sequence\ like\ ABC \tag{12}$$

Note that in this protoform P_S is used instead of P. This is to make clear that the summarizer focuses on a sequence instead of the complete case. This protoform can be used to get a high level overview of attributes within a feature, e.g. *"In most cases, there was a large throughput time for a sequence like <Send advice, Create dossier, Process decision>"* or *"In most cases, there were many different resources for a sequence like <Send advice, Create dossier, Process decision>"*.

Since there is no qualifier used in this sentence, the degree of focus is equal to one. However, the sentence is only about cases that contain a sequence like ABC, and, therefore, the sequence frequency [24] is used, instead of the degree of focus, to specify the amount of times a sequence was present in the event log. The calculation of the sequence frequencies of all protoforms (where available) can be found in Appendix C.

Also for the sequence focused protoforms a qualifier can be used to get an indication of the number of cases a sequence was P_S, when the condition is fulfilled [23]. The extended, sequence focused protoform is written as:

$$In\ Q\ cases, when\ condition\ R\ was\ fulfilled, there\ was\ P_S\ for\ a\ sequence\ like\ ABC \quad (13)$$

An example sentence related to protoform (13) is *"In most cases, when there was a large throughput time, a sequence like <Send advice, Create dossier, Process decision> was performed by Person A"*.

However, similar to protoform (9) it might be interesting to set the sequence as the condition. By doing so, two more protoforms are relevant:

$$In\ Q\ cases, when\ there\ was\ R_S\ for\ a\ sequence\ like\ ABC, there\ was\ P \quad (14)$$

$$\begin{gathered} In\ Q\ cases, when\ there\ was\ R_S\ for\ a\ sequence\ like\ ABC, \\ there\ was\ P_S\ for\ a\ sequence\ like\ XYZ \end{gathered} \quad (15)$$

Protoform (14) provides information about the complete case, but has a sequence focused aspect as the condition. This protoform creates sentences like *"In most cases, when there was Person A for a sequence like <Send advice, Create dossier, Process decision>, there was a large throughput time"*. Note that both the sentence and the information represented is different than *"In most cases, when there was a large throughput time, there was Person A for a sequence like <Send advice, Create dossier, Process decision>"*. Protoform (15) can be used to get indications which sequence is correlated other sequences.

Note: This thesis focuses on clusters of sequences instead of matching one particular sequence. When one is interested in matching specific sequences, clusters can be created of maximal size one. All protoforms and statistics remain the same, but the word *"like"* can be removed. For that reason, the exact sequence is not elaborated further in this section.

3.3 Other relevant protoforms

In Section 3.1 and 3.2 the case focused and sequence focused protoforms are discussed. However, more protoforms may be relevant for the summarization of process data. In this section some other protoforms are described. These protoforms focus, for instance, on temporal aspects (Section 3.3.1), the use of gradual rules (Section 3.3.2), and what patterns might be relevant (Section 3.3.4). Nevertheless, these protoforms are not discussed further in this master thesis.

3.3.1 Temporal aspects

The protoforms mentioned before refer to the complete process. However, since event logs contain timestamps, one can use this data to see whether the process changed over time. In [38] research is performed about linguistic summarization of time series, where the temporal aspect is expressed with E_T. Two protoforms are discussed in [38]:

$$E_T\ among\ all\ y's, Q\ are\ P \tag{16}$$
$$E_T\ among\ all\ R\ y's, Q\ are\ P \tag{17}$$

Using these type of protoforms, sentences like *“recently among all cases, most are long”* or *“recently among all cases where Person A was involved, almost all are sent on”* can be created. By using the temporal expression, one can investigate a certain time period of the event log, to determine whether there were changes over time. The second example sentence might indicate that *Person A* simply sends on all cases recently. This might need further investigation.

3.3.2 Gradual rules

Gradual rules (described in [39] and [40]) can also help in analyzing processes, to detect what attributes have got influence on other attributes. Gradual rules are of the form:

$$The\ more\ X\ is\ F, the\ more\ Y\ is\ G \tag{18}$$

Where *“the more”* can be replaced by *“the less”*. By use of gradual rules, sentences like *“The larger the throughput time, the shorter the waiting time”* or *“The shorter the waiting time, the more resources are used”* can be created.

3.3.3 Compare (similar) processes

Section 3.3.1 shows how temporal expressions help analyzing processes. However, one might not be interested in how the process evolved over time, but wants to compare it with another (maybe older) process. Examples of protoforms that are related to the comparison of processes are:

$$In\ Q\ cases, there\ was\ P\ in\ process\ X \tag{19}$$
$$In\ Q\ cases, where\ condition\ R\ was\ fulfilled, there\ was\ P\ in\ process\ X \tag{20}$$

Note that these protoforms are highly related to protoforms (6) and (7), where the protoforms are extended with *“in process X”*. This can also be applied for other protoforms discussed in this chapter. Protoform (19) can create sentences like *“In most cases, there were more resources involved for the new appeal process”*. When protoform (20) is used, sentences like *“In almost all cases, where sequence ABC was present, there was a larger throughput time for the new appeal process”* can be created. However, one has to be careful with selecting the features that are evaluated for these protoforms. A sentence like *“In most cases, where the throughput time was large, Person A was involved in the old appeal process”* would probably not make sense, since other resources may be involved in the other (newer) process.

3.3.4 Patterns

As discussed in Section 3.2, focusing on sequences instead of on the complete case can give insight about certain parts of a process. A sequence is a set of events that occurs in a certain order. However, as discussed in [22], a distinction can be made between directly succeeding actions or allowing other in-between actions. New protoforms can be created when allowing in between actions. All protoforms discussed before, related to sequences, can be transformed such that they allow for other, in-between actions. Examples are:

$$\text{Q cases contain a sequence A} * \text{B} * \text{C} \tag{21}$$
$$\text{In Q cases, there was } \mathrm{P_S} \text{ for sequence A} * \text{B} * \text{C} \tag{22}$$
$$\text{In Q cases, when condition R was fulfilled, there was } \mathrm{P_S} \text{ for sequence A} * \text{B} * \text{C} \tag{23}$$

In these protoform * is used to indicate any number of activities.

Other patterns that might add value to the summarization is parallelism vs sequentialism. One may be interested in the influence of parallel or sequential performed activities. activities performed parallel might improve the process. An example sentence is *"The more activities ABC are done in parallel, the shorter the throughput time"*. For this kind of sentences, the gradual protoform (18) can be translated to:

$$The\ more\ activities\ ABC\ are\ done\ F, the\ more\ Y\ is\ G \tag{24}$$

In this case "*F"* is either parallel or sequential and *"the more"* can be changed with *"the less"*.

The last pattern discussed in this section is the iterative pattern. One might be interested why certain sequences of actions are performed multiple times, or what the influence of cycles is on the process. Examples of protoforms are:

$$In\ Q\ cases, sequence\ ABC\ was\ cyclic \tag{25}$$
$$In\ Q\ cases, when\ condition\ R\ was\ fulfilled, sequence\ ABC\ was\ cyclic \tag{26}$$
$$In\ Q\ cases, when\ sequence\ ABC\ was\ cyclic, there\ was\ P \tag{27}$$

The simple protoform (25) can be used to get a basic understanding of the process. The extended protoform (26) can be more useful when one wants to investigate why a cycle occurred. When a sentences like *"In most cases, when Person A was involved in the process, sequence ABC was cyclic"* is generated, one can choose to investigate whether *Person A* is making mistakes. To see the influence of cycles on the rest of the process, the other extended protoform (27) can be useful. This protoform is used to create sentences like *"In most cases, when sequence ABC was cyclic, Withdraw appeal is performed"*.

3.3.5 Fuzzy matching between attributes

[41] introduces fuzzy matching between attributes. Sentences like *"In most cases, there was a large throughput time"* and *"In some cases, there was a short throughput time"* can be combined by using fuzzy matching. The resulting sentence then looks like *"In most cases, there was a large throughput time, but for some there was a short throughput time"*. A protoform related to this can be written as:

$$In\ Q_1\ cases, there\ was\ P_1, but\ for\ Q_2\ cases, there\ was\ P_2 \tag{28}$$

In this protoform Q_1 <> Q_2 and P_1 is related to P_2.

3.3.6 Usuality

[42] investigates the pessimistic and optimistic probabilities that can help in the summarization of data. Sentences that are related to this are *"Usually, the throughput time of cases is large"* for the optimistic probability and *"Rarely, the throughput time of cases is large"* for the pessimistic one.

As discussed in [22], this topic needs additional research. All protoforms discussed above can be transformed to this type, where Q is either usually or rarely, e.g. protoforms (6) and (7) can be converted to:

$$Usually/Rarely, there\ is\ P \tag{29}$$

$$Usually/Rarely, \text{when condition R was fulfilled, contain a sequence like ABC} \tag{30}$$

3.3.7 Trends

The last protoform discussed in this research is based on [43] and [44]. Event logs can contain much data, distributed over a long period of time. One might be interested in finding some trends in the data that occur frequently. The protoform introduced is written as:

$$M\ every\ p\ unit, the\ data\ take\ high\ values \tag{31}$$

Where M is an adverb, such as exactly or approximately. This protoform is used to create sentences like *"Exactly every 2 months, Sent on occurs more often"*.

3.4 Combinations of protoforms

Many protoforms are introduced in this chapter. However, the summarizer P and qualifier R of these protoforms focus on one single aspect, e.g. on the throughput time or on the occurence of a sequence. Sentences can also be combined to include several aspects, e.g. *"Most cases contain a sequence like ABC and have got a large throughput time"* can be created when combining protoform (6) and (8). When doing this, more information can be included in the same sentence. In this thesis, summarizers and qualifiers are not combined, because of the running time and because this topic needs further investigation.

3.5 Protoforms in scope

This research focuses on the case focused protoforms described in Section 3.1 and the sequence focused protoforms explained in Section 3.2. This choice is made because they are the most general cases for event logs and are assumed to be relevant for most processes. The protoforms that are investigated further are:

$Q\ y's\ are\ P$

- Protoform (6): In Q cases, there was P;
- Protoform (8): Q cases contain a sequence like ABC;
- Protoform (12): In Q cases, there was P_S for a sequence like ABC;

$Q\ R\ y's\ are\ P$

- Protoform (7): In Q cases, when condition R was fulfilled, there was P;
- Protoform (9): Q cases, when condition R was fulfilled, contain a sequence like ABC;
- Protoform (10): In Q cases, that contain a sequence like ABC, there was P;
- Protoform (11): Q cases, that contain a sequence like ABC, contain sequence like XYZ;
- Protoform (13): In Q cases, when condition R was fulfilled, there was P_S for a sequence like ABC;
- Protoform (14): In Q cases, when there was R_S for a sequence like ABC, there was P;
- Protoform (15): In Q cases, when there was R_S for sequence like ABC, there was P_S for a sequence like XYZ.

Concluding, three protoforms are constructed, based on the simple protoform (1), namely by: 1) relating it with cases, 2) looking at the containment of sequences, and 3) analyzing sequences in more depth. Combining these topics in the summarizer and qualifier results in seven new protoforms based on the extended protoform (2).

4 Process flow

In this chapter, all steps that need to be performed to create sentences about an event log are discussed in a high level overview. Figure 4 visualizes the process of deriving sentences from an event log, where blue steps are mostly based on user interaction, green steps can be performed fully automatic, and yellow steps only need some user interaction, but are mainly automatic. The figure also includes information which chapter or section discusses the steps further. Since every event log is unique, the execution of these process steps can be slightly different, e.g. different features are analyzed and different parameters need to be set.

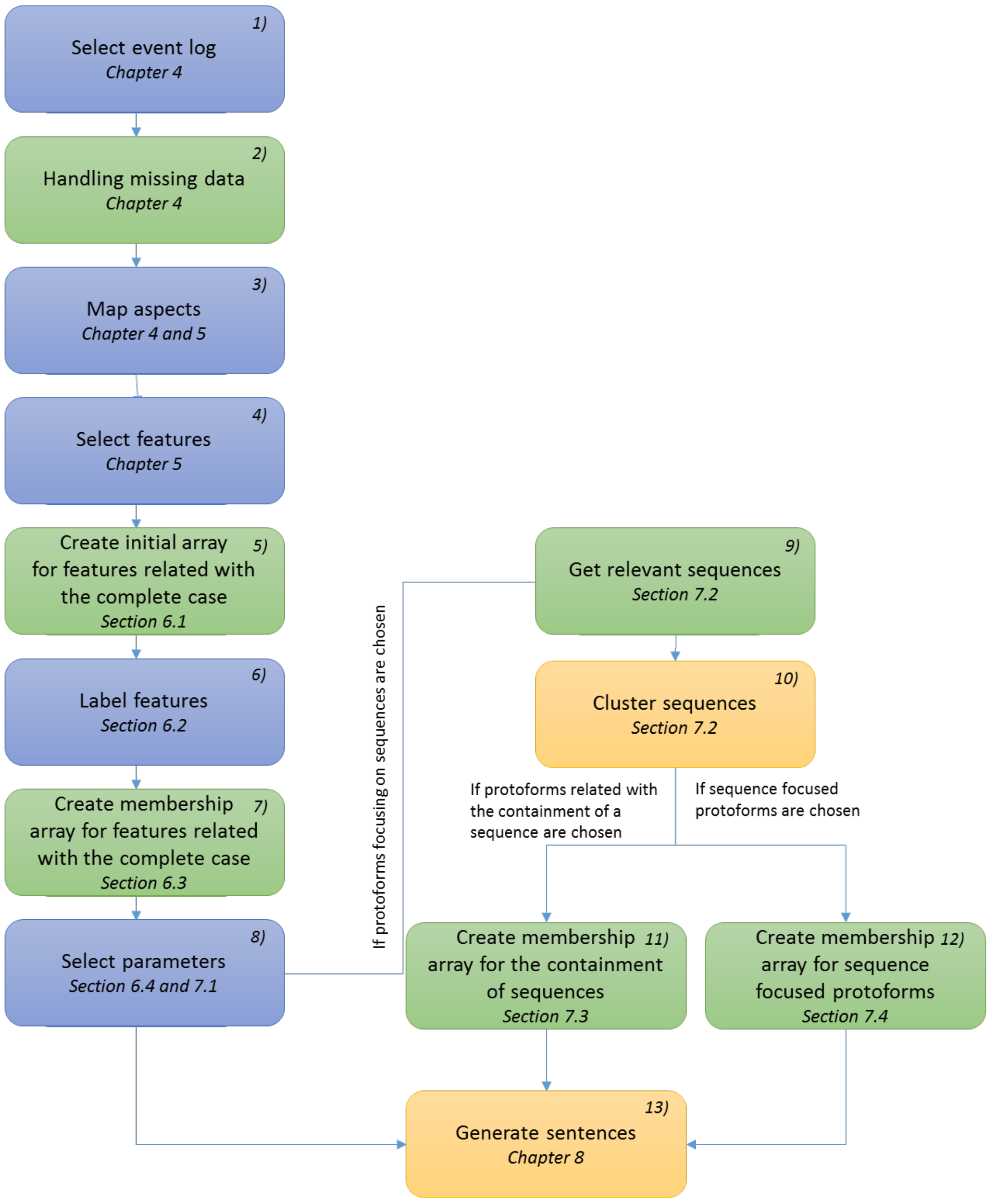

1) Select event log *Chapter 4*
2) Handling missing data *Chapter 4*
3) Map aspects *Chapter 4 and 5*
4) Select features *Chapter 5*
5) Create initial array for features related with the complete case *Section 6.1*
6) Label features *Section 6.2*
7) Create membership array for features related with the complete case *Section 6.3*
8) Select parameters *Section 6.4 and 7.1*
If protoforms focusing on sequences are chosen
9) Get relevant sequences *Section 7.2*
10) Cluster sequences *Section 7.2*
If protoforms related with the containment of a sequence are chosen
If sequence focused protoforms are chosen
11) Create membership array for the containment of sequences *Section 7.3*
12) Create membership array for sequence focused protoforms *Section 7.4*
13) Generate sentences *Chapter 8*

Figure 4: Process flow

1) **Select event log**

   First, the event log should be selected. For some event logs, no timestamps are logged, or activities are performed simultaneously, e.g. when a form is filled and all information is stored by use of a button, both *Create fine* and *Set amount* are logged at the same time. When the start timestamp of multiple activities is the same, or if no timestamp is logged, it cannot be determined automatically what activity should be performed as first action. In the example shown above, it is expected that *Create fine* occurs before *Set amount*. To make sure that the correct process flow is analyzed, the ordering is not done automatically, but is left as the responsibility of the user.

2) **Handling missing data**

   Data can be missing in the event logs. [45] discusses various techniques how to handle missing data. Since the algorithm is for event logs in general, it is hard to identify missing data. Missing data can for example be indicated with NULL, by use of an empty cell, or by 0. However, for other processes, these values can actually be valid values. Also missing data should be treated differently for different processes: for some processes it can be deleted and for other processes the values have to be interpolated. For the reasons mentioned above, this data is not changed, and empty cells are converted to NULL. In this way, also sentences can be created like *"In most cases, when Person A was present, the outcome was NULL"*. In this thesis it is assumed that the case identifier and the timestamps, when available, do not contain missing data, i.e. they are always filled.

3) **Map aspects**

   An event log can contain a couple of different aspects, namely: A case identifier, an activity that is performed, a start timestamp, an end timestamp, the transactional life cycle, one or multiple case attributes and one or multiple activity attributes. The different aspects are further explained in Chapter 5. Since every aspect needs to be treated differently and it is difficult to identify what is represented by every column automatically, the user has to define this.

4) **Select features**

   Based on the available data, different features can be calculated, e.g. when there is only one timestamp, no operating time can be calculated. This topic is further described in Chapter 5.

5) **Create initial array for features related to the complete case**

   After the features are selected, all relevant information is stored that is needed to analyze every feature that is related to the complete case. More information about the generation of this initial array is given in Section 6.1.

6) **Label features**

   Some features do not need any user input to be analyzed, e.g. *Sent on* was performed or *Person A* was present can be based on data only. However, to be able to create sentences such as *"In most cases, there was a large throughput time"*, the meaning of a *large throughput time* must be known. To deal with such features, labels need to be defined by the user, which is further explained in Section 6.2.

7) **Create membership array for features related to the complete case**
   Once all labels are defined, the array created in Step 5) can be converted into a membership array, e.g. when the user created a membership function for a *large throughput time*, the values of the throughput time are replaced by the membership values of that feature. There are many features that all need a different conversion, which is explained in Section 6.3.
8) **Select parameters**
   Before sentences can be created, some parameters have to be defined, e.g. the protoforms that the user wants to analyze or the degree of focus. The parameters that are relevant for all protoforms are explained in detail in Section 6.4. Additional parameters, that are relevant when sequences are analyzed, are discussed in Section 7.1.
9) **Get relevant sequences**
   If the user wants to create sentences focusing on sequences, all sequences that are present in the dataset need to be identified. Since there may be many unique sequences, sequences are filtered on multiple aspects. This is further described in Section 7.2.
10) **Cluster sequences**
    Even when sequences are filtered, the remaining ones can be numerous. One can choose to group similar sequences and create sentences focusing on the complete group. Such a group of sequences is also called a cluster. The techniques that are used to cluster sequences are highlighted in Section 7.2.
11) **Create membership array for the containment of sequences**
    Once clusters of sequences are created, new features can be analyzed. One possible type of feature is the containment of a cluster, to create sentences like *"Most cases contain a sequence like <Send advice, Create dossier>"*. To create such sentences, a new membership array is created that stores what cluster of sequences is contained in what case. The construction of this array is explained in Section 7.3.
12) **Create membership array for sequence focused protoforms**
    Next to the containment of sequences, one can choose to analyze sequence focused protoforms, to create sentences like *"In most cases, where there was Person A for a sequence like <Send advice, Create dossier>, there was a large throughput time"*. In this case, another membership array is needed where the features related to the sequence focused protoforms are stored. The creation of this membership array is explained in Section 7.4.
13) **Generate sentences**
    When all steps are executed, sentences can be created, based on the event log. As input for the analysis, this step needs the protoforms and the corresponding membership array(s). As output, sentences are created that are based on the available data. The generation of the sentences is discussed in Chapter 8.

5 Analysis of possible features

There are many features that may be relevant for the analysis of process data. In this chapter, it is determined what features can be relevant, such that most aspects can be analyzed for most event logs. The set of features is not complete, and, therefore, the algorithm is created such that features can be extended easily. Feature are included based on the structure of an event log. In [1], it is assumed that an event log contains data related to a single process and each event refers to a single process instance, which is referred to as the case. For this research, the same assumptions are used. Every event in the event log is related to an activity. The case identifier and activities performed are considered as the bare minimum that should be present in an event log. This chapter first introduces the features that are associated with the bare minimum in Section 5.1. Section 5.2 describes what other features may be relevant when some form of a lifecycle of activities is added to the event log. Section 5.3 explains the influences of other attributes, e.g. resources and costs. Besides looking at all aspects of an event log (columns) in isolation, columns can be combined, which is described in Section 5.4. Finally, the features selected in scope of this research are summarized in Section 5.5.

5.1 Bare minimum of features available

As described above, the bare minimum of an event log consists of two columns, namely: 1) a case identifier and 2) an activity. All features depend on the case identifier. For example, to determine the throughput time of a case, the case identifier is needed to recognize which times should be summed. Since the activities are ordered within every case, sequences can be investigated. The features that can be relevant focusing on activities are:

- Whether a certain activity is performed in a case;
- Number of times an activity is performed in a case;
- Number of (distinct) activities performed in a case;
- Whether an activity is performed as first or last event;
- Whether a certain sequence is present in a case (as discussed in Section 3.1, this feature is investigated in a separate protoform).

5.2 Lifecycle of activities

Every case of an event log is already ordered on time. However, more information can be extracted by adding the lifecycle of activities to the event log. This lifecycle can be logged by use of a transactional life-cycle [1], by adding start and/or end timestamps, or by a combination of both. In this section, the features, that might be relevant to analyze when the lifecycle of activities is logged, are described.

5.2.1 Transactional life-cycle model

Figure 5 shows a graphical representation of the transactional life-cycle model [1]. One may be interested in investigating when an activity has been reassigned, suspended, skipped, unsuccessfully aborted, or successfully aborted. Therefore, this data can be added to the set of features. The transactional life-cycle can also be present partially, i.e. only the start and end timestamp of the activities are measured. Since the transactional life-cycle contains information about the lifecycle of activities, also certain patterns in the event log, like parallelism vs sequentialism, can be investigated.

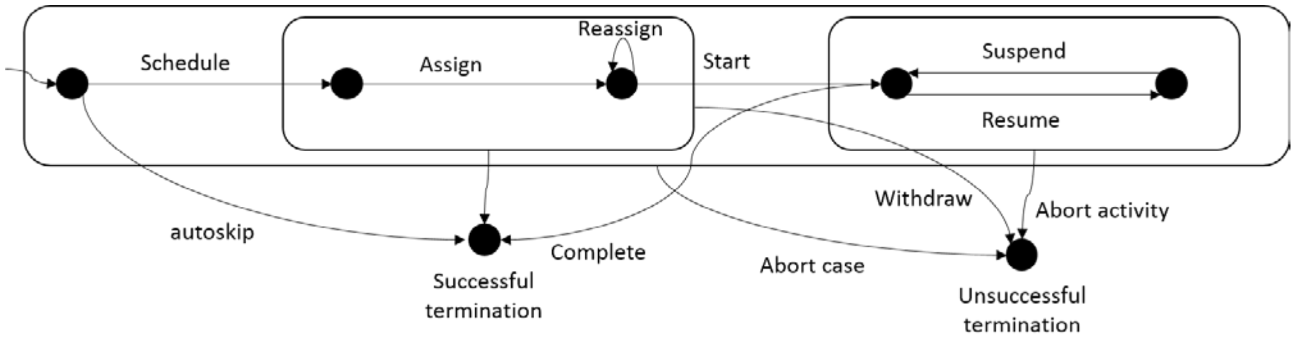


Figure 5: Standard transactional life-cycle model [1]

For the scope of this research, no further research is done in the transactional life-cycle model, since it can cause many problems. Figure 6 shows an example of a potential problem. In this scenario, first *Send advice* is started. Before the activity is finished, another instance of *Send advice* is started for the same case. Since two instances of the activity are started, there also exist two end instances. In this case, it is not clear what time is linked to what instance [1]. In Figure 6, two possible scenarios are shown how the end stages can be linked to the start stages (represented with the dotted line). However, the operating times of the activities differ (the first scenario has operating times of 3 and 1 days, for the other scenario, both operating times are equal to 2 days), which can impact the analysis. The scenario can be much more difficult when considering multiple stages of the transactional life-cycle model, or if more instances of the same activity are started. Since the transactional life-cycle is not investigated in this thesis, one can either set it as an attribute, or filter all rows except for the end indication of the activity. In this way, the process can still be analyzed partially.

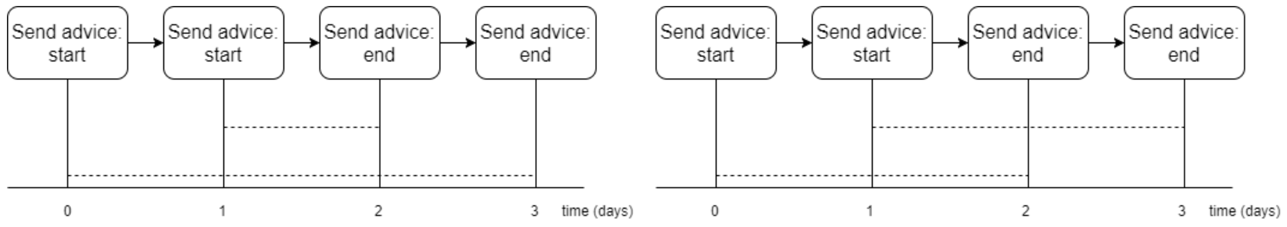


Figure 6: Scenario two activities same label

5.2.2 Timestamps in an event log

The second way of adding a lifecycle of activities is by adding start and/or end timestamps for every activity, in separate columns. Also if the lifecycle of activities is added in this manner, a scenario similar to Figure 6 can occur. For the same reasons as mentioned before, this scenario is not investigated in this thesis, and can, therefore, not be analyzed with the current status of the algorithm. This kind If both the start and the end timestamp are logged, the following features can be added:

- Throughput time of a case / sequence;
- Waiting time of a case / sequence;
- Operating time of a case / sequence / activity.

If only one timestamp is logged, only the *throughput time* of the case and sequences can be measured. The *operating time* is equal to zero, and, therefore, the *waiting time* is equal to the *throughput time*. Definitely, the *operating time* is not zero in real-life, therefore, the *throughput time* can be seen as *waiting time + operating time*.

5.3 Attributes in an event log

In this research, a distinction is made between two types of attributes, namely: 1) case attributes and 2) activity attributes (also called event attributes). Case attributes are attributes that remain the same during the complete case, e.g. the outcome of the case. activity attributes are attributes that are local to the events and they can get different values at different events (even within a single case), e.g. the costs associated to an activity or the resource that performed an activity. The different types of attributes are handled differently.

Case attributes

A case attribute is an attribute that remains the same during the complete case. Different features can be relevant, depending on the (number of) values that are associated with the attribute. If there is a limited number of options, e.g. whether the case is about a certain permit or whether compensation for the school bus is given, one can keep track of the value that is set (e.g. compensation for school bus). In the remainder of this thesis, these kind of attributes are referred to as limited attributes. However, it may be the case that there is an unlimited number of different options available for the attribute, e.g. the costs associated with the complete case can be any number (i.e. it is a continuous value). It may be relevant to define certain groups of values, for example, whether the costs are high, medium or low. These kind of attributes are referred to as unlimited attributes in the remaining of this thesis. This can be done by introducing membership functions for the attribute for the different groups, and is explained in Section 6.2.

Activity attributes

An activity attribute is an attribute of which the value can change during the lifecycle of a case. The same features are relevant as the features described for the case attributes. However, if there are a limited number of distinct values, other relevant features may be:

- Number of distinct values set in a case / sequence;
- Number of times a certain value is set in a case / sequence.

The features above can be used to create sentences like *"In most cases, there were many resources present"* or *"In most cases, Person A was present many times for the sequence <Send advice, Create dossier>"*, where both features are based on membership functions. However, the membership functions can be based on either frequencies or percentages of their occurrence, e.g. "*there were many resources if there were more than 2 distinct resources present"* for the frequency based version, or "*there were many resources if at least 70% of the activities is performed by different resources"* for the percentage based version. When the frequency based version of the features have to be evaluated for sequences, the user must define a membership function for every sequence. This is since the length of sequences may differ.

Different types of activity attributes

Not all activity attributes are handled in a similar way. If one wants to analyze, for example, the column related to the resources, the features explained above can be calculated, e.g. *Person A was present*, *Person A was present many times* or *there were many distinct resources*. However, imagine a column that specifies a certain amount of a fine. The amount can be initialized at the generation of the fine. If the fine is not paid, an amount can be added in an activity *Add penalty*. This is visualized in Table 1. The amount related to *Add penalty* can either be the amount that the fine is raised, as in Table 1, or the new amount (€130 in the example case).

If the amount has a different meaning, the user may want to specify different membership functions for the different activities and analyze them separately, e.g. the amount is high if it is higher than €100 for *Create fine*, but it is high if it is above €30 for *Add penalty*. However, it can be the case that the user wants to evaluate all activities in isolation, but the membership function is the same for all activities. By looking at all activities in isolation, sentences can be created like *"In most cases the amount was high for add penalty"*.

The user does not necessarily want to evaluate all activities in isolation, e.g. if the amount is identified with the same membership function, the user might only want to see that the amount is high, but is not interested in a specific activity.

Table 1: The value of the activity 'Amount' varies within a case

Case ID	*Activity*	*Amount (€)*
1	Create fine	100
1	Send fine	
1	Add penalty	30

Membership functions are based on numbers, e.g. the costs are high when they are above €100. However, a value of an attribute can be a string, date, integer, float or boolean [1]. Therefore, when there is an unlimited number of options for an attribute that is based on strings or dates, no membership function can be created. For this reason, unlimited attributes that are not based on numbers, are not in scope for this thesis.

5.4 Combinations of features

There are three kind of perspectives related to event logs, namely: 1) process perspective, 2) case perspective, and 3) organizational perspective [46]. The process perspective focuses on the control flow, i.e. certain sequences in the process, the case perspective is related to the case attributes, and the organization perspective is related to the activity attributes. As described in [46], the different perspectives are often highly related and cannot be seen in isolation. Therefore, it may be relevant to analyze a combination of columns, i.e. link the resources to the activities to analyze who performed what activity. Note that a combination can contain more than two columns, i.e. not only link the resources to the activities but also include the manager that was present.

Limitations

Both columns with an unlimited or limited number of values can be linked. An example of a sentence that may be relevant for a limited number of values is *“In most cases, where Sent on was associated with Person A, there was a large throughput time”*. An example of a sentence that might be relevant for an unlimited number of options is *“In most cases, where Person A was associated with high costs, there was a large throughput time”*. For the scope of this project, it is chosen to only focus on limited values, since the other case is less likely to be useful. However, the program is designed in such a way that unlimited attributes can be included easily, by implementing a method that calculates the fuzzy membership of the combinations.

It is difficult to select the combination of columns automatically, since different combinations are relevant for different purposes. There can be many columns, so looking at all combinations can result in both a gigantic set of results that is not useful for the user and a gigantic running time. Therefore, it may be more efficient if the user selects the combinations that are relevant. By doing so, only relevant combinations are analyzed and the user is not overloaded with results.

5.5 Relevant features included in the analysis

All features that are selected for this research are summarized in Table 2. In the first column, the name of the feature is stated. In the second column, the minimal requirements of columns that should be present in the event log, is stated. In the third column, the method used to measure the feature is filled. There are two options for this column:

- Membership function: A membership function needs to be defined to specify to what fuzzy set the value belongs, e.g. *“a large throughput time”* or *“high costs”*;
- Boolean: No membership function has to be defined. The feature can be expressed by a boolean, e.g. the features *“Activity = Register appeal”* or *“Resource = Person A”* are either true or false.

Note that the combination of columns is defined with a boolean. It may, however, be the case that one or more columns that are used to create the combination are defined by a membership function, but this case is not covered in this research. In the last column of Table 2, an example feature is shown. All features that do not focus on sequences are considered as features related to the complete case.

Table 2: Features in scope of this research

Feature	Requirements[1]	Estimated with	Example
Activity performed		Boolean	The activity ‘Send on’ is performed
Number of times an activity is performed		Membership function	The activity ‘Send on’ is performed many times
Number of activities		Membership function	Large number of activities

[1] The bare minimum, as described in Section 5.1, should always be present and is, therefore, not shown as requirement.

Feature	Requirements[1]	Estimated with	Example
Number of distinct activities		Membership function	Large number of distinct activities
First activity		Boolean	The activity 'Register appeal' is performed as first activity
Last activity		Boolean	The activity 'Archive' is performed as last activity
Throughput time case	One timestamp	Membership function	Large throughput time
Throughput time sequence	One timestamp + sequence	Membership function	Large throughput time for a sequence like <Process decision, Create dossier>
Waiting time case	One timestamp	Membership function	Short waiting time
Waiting time sequence	One timestamp + sequence	Membership function	Short waiting time for a sequence like <Process decision, Create dossier>
Operating time case	One timestamp + end time	Membership function	Average operating time
Operating time sequence	One timestamp + end time + sequence	Membership function	Average operating time for a sequence like <Process decision, Create dossier>
Operating time activity	One timestamp + end time	Membership function	Average operating time for the activity 'Send on'
Value selected – unlimited case attribute	Unlimited case attribute	Membership function	Low risk (where risk is between 0 and 1)
Value selected – limited case attribute	Limited case attribute	Boolean	Risk = low (where risk is low, medium or high)
Value selected – unlimited activity attribute	Unlimited activity attribute	Membership function	Large costs
Value selected – limited activity attribute	Limited activity attribute	Boolean	'Person A' was involved
Number of distinct values selected, frequency based	Limited activity attribute	Membership function	Large number of distinct resources
Number of distinct values selected, percentage based	Limited activity attribute	Membership function	Large number of distinct resources
Number of distinct values selected for sequences	Limited activity attribute	Membership function	Large number of distinct resources for a sequence like <Process decision, Create dossier>
Number of times a value is selected, frequency based	Limited activity attribute	Membership function	'Person A' is selected many times
Number of times a value is selected, percentage based	Limited activity attribute	Membership function	'Person A' is selected many times
Number of times a value is selected for sequences	Limited activity attribute	Membership function	'Person A' is selected many times for a sequence like <Process decision, Create dossier >
Combination of columns	Limited attribute	Boolean	Person A performed the activity 'Archive'

6 Implementation issues concerning features related to the complete case

Chapter 5 discusses what features can be relevant for process data. If someone wants to analyze, for example, the throughput time, it has to be calculated for all cases. This data is needed every time a sentence is created about the throughput time, and, therefore, it may be efficient to store all throughput times in a separate variable. In this chapter, it is discussed in what manner the data can be stored in an efficient way such that the algorithm does not have to go through the complete dataset over and over again. This chapter focuses on features related to the complete case. Section 6.1 discusses how all relevant data is stored. However, the generation of sentences is based on a membership array. To convert the initial array to this membership array, linguistic labels need to be defined for certain features, explained in Section 6.2. The generation of the membership array is discussed in Section 6.3. Last, in Section 6.4, some parameters are discussed that can be used to scope the analysis, such that only relevant aspects are analyzed. Finally, Section 6.5 concludes wat can be achieved with the logic described in this chapter.

6.1 Create initial array

In the first step, a $n\ x\ m$ array is created, where n is the number of cases and m is the amount of features that may be relevant. In cell$_{i,j}$, all information about case *i* is stored that is needed to analyze feature *j*. In Table 3, an example of an initial array is shown. In this array, the throughput time of a case, whether the activity *Sent on* is performed, and the operating time of *Sent on* are logged. The features shown in Table 3 are all handled differently. A distinction is made between three kinds of values:

- Boolean: In this case the feature can be stored as either True or False for every case. An example is shown in the second column of Table 3, where *Send on* is either performed or not;
- Value: In this case a membership function is needed to convert the data to the membership values. See for example column three of Table 3. For this feature, membership functions need to be defined to create several sets for the throughput time, e.g. short, average, and large;
- List: These features can have different values for the same case, or may have no values at all. An example is shown in the fourth column of Table 3. In this column, the operating time of *Send on* is logged. This activity may not be present, which results in an empty list (e.g. case 2 or 3). It can also be the case that the activity is present once (e.g. case 4) or more than once (e.g. case 1). In these cases, the list contains the values of every occurrence.

Table 3: Example initial array

Case ID/feature	*"Send on" performed*	*Throughput time case*	*Operating time "Send on"*
1	True	100	[30, 40]
2	False	30	[]
3	False	1	[]
4	True	50	[40]
…	…	…	…

Table 4 states how all case focused features should be handled (column one and two), where in the third column an explanation is given, e.g. a formula or why a certain value is chosen. This table consists of a subset of the features introduced in Table 2.

Table 4: How to handle case focused features

Feature	How to store	Explanation
Activity performed	Boolean	Activity was present or not.
Number of times an activity is performed	Value	Number of times an activity is performed is logged per activity.
Number of activities	Value	Number of activities performed in a case.
Number of distinct activities	Value	Number of distinct activities performed in a case.
First activity	Boolean	Activity was executed as first event or not.
Last activity	Boolean	Activity was executed as last event or not.
Throughput time case	Value	$Throughput\ time\ = start\ time - end\ time.$
Waiting time case	Value	$Waiting\ time = Throughput\ time - Operating\ time.$
Operating time case	Value	$Operating\ time =$ $\sum_{i=1}^{n} end\ time\ activity_i - start\ time\ activity_i$, where n is the number of activities in the case.
Operating time activity	List	Activity can be performed multiple times.
Value selected – unlimited case attribute	Value	Case attribute is the same for the complete case.
Value selected – limited case attribute	Boolean	A certain value was logged or not.
Value selected – unlimited activity attribute	List	Multiple values for the activity attribute can be logged for the same case.
Value selected – limited activity attribute	Boolean	A certain value was logged or not.
Number of distinct values selected, frequency based	Value	Number of distinct values that are logged for the corresponding attribute for that case.
Number of distinct values selected, percentage based	Value	Number of distinct values that are logged for the corresponding attribute for that case / number of events within the case.
Number of times a value is selected, frequency based	Value	Number of times a value is logged for the corresponding attribute for that case.
Number of times a value is selected, percentage based	Value	Number of times a value is logged for the corresponding attribute for that case / number of events within the case.
Combination of columns	-	Is done in a later stage to include the possibility to add unlimited attributes.

6.2 Linguistic label definitions

To be able to create sentences like *"In most cases there was a large throughput time"*, linguistic labels need to be defined, e.g. a *large throughput time*. In this thesis, trapezoidal functions are used to define the fuzzy sets, which are commonly used to model fuzzy sets [47] and are easy to understand. A trapezoidal function is defined with five parameters: a *label*, *point a*, *point b*, *point c* and *point d*. These points indicate the boundaries of the trapezoidal function.

An example of a trapezoidal function is shown in Figure 7. It can be seen that values smaller than *point a* or values greater than *point d* lead to 0 membership. Values between *point b* and *point c* lead to a membership of 1 and other values lead to a membership between 0 and 1, summarized in the following formula [48]:

$$f(x) = \begin{cases} 0, & x \leq a \\ \dfrac{x-a}{b-a}, & a \leq x \leq b \\ 1, & b \leq x \leq c \\ \dfrac{d-x}{d-c}, & c \leq x \leq d \\ 0, & d \leq x \end{cases} \tag{32}$$

This formula can also be written as:

$$f(x; a, b, c, d) = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right) \tag{33}$$

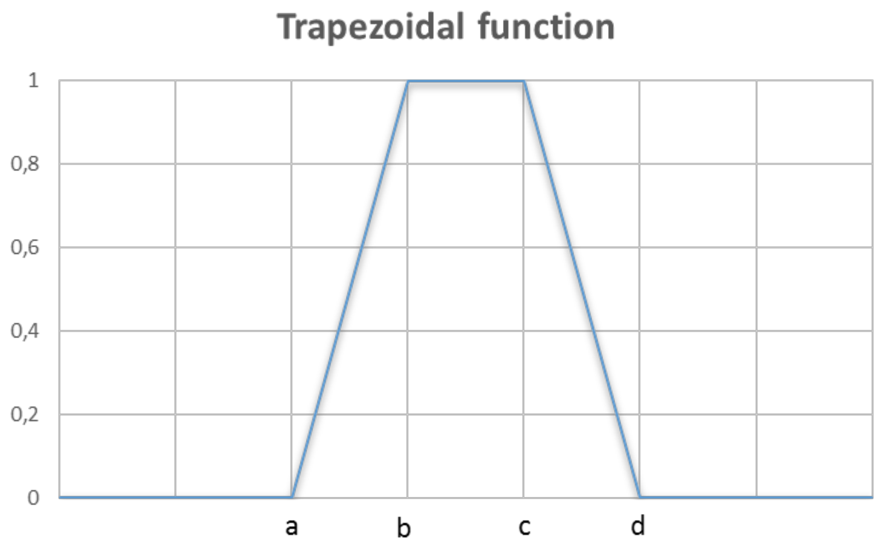


Figure 7: Trapezoidal function

In the scenario shown above, all points have unique values. However it can be the case that one or more points overlap. Some possible cases are shown in Figure 8. If *point b* = *point c* (upper left case), a so-called triangular function is obtained, where only one specific point returns a membership of 1. This could be used for labels like ‘Around 3’. If *point c* = *point d* (upper right case), a specific upper bound is set and if *point a* = *point b* (left bottom case), a specific lower bound is set. The last case is if *point a* = *point b* and *point c* = *point d* (right bottom case), where both an upper and lower bound are crisp.

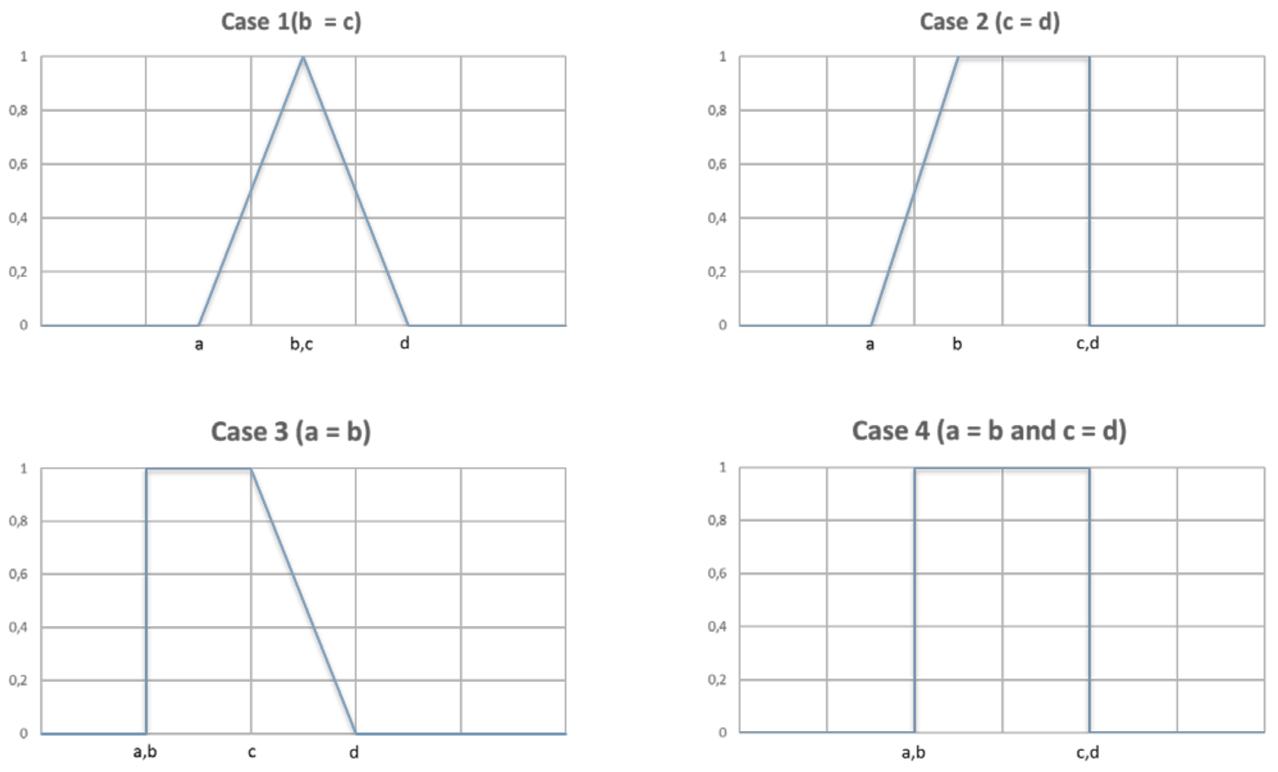


Figure 8: Trapezoidal function cases

Many times, multiple membership functions are used for a single feature, e.g. *short*, *average* and *large* for the throughput time. An example setting of membership functions for the throughput time is shown in Figure 9. It can be seen that the throughput time is, for example, *short* with a membership of 1 if the throughput time is in between 0 and 2 and *average* with membership of 1 when it is in between 4 and 7. If the throughput time is for example 3, it is *short* with a membership of 0.5, *average* with a membership of 0.5 and *large* with a membership of 0.

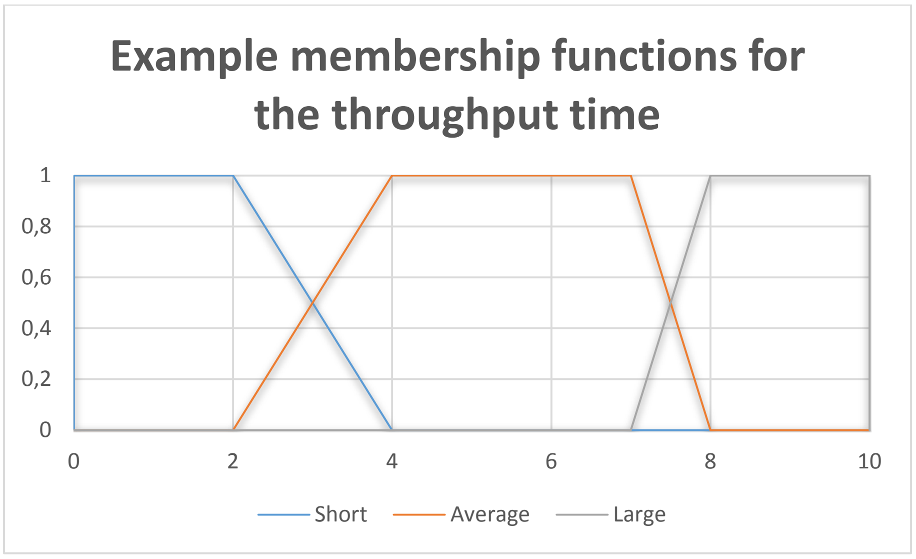


Figure 9: Example of membership functions for the throughput time

6.3 Create membership array using linguistic labels

After all linguistic labels are defined, the membership array is created. The membership array is a $n\ x\ \sum_{i=1}^{m}(m_i * \#mf_i)$ array, in which n is the number of cases and $\#mf_i$ the number of membership functions defined for feature m_i. For example, if the membership functions short, average and large are defined for the throughput time, these are all stored as separate, new, features. The membership array must only contain the membership, of every feature, to each case. To achieve this, the initial membership is converted by use of the membership functions. The three kind of values defined in Section 6.1 are all handled differently:

- Boolean: These values already contain the membership and can, therefore, be copied into the membership array without any transformation;
- Value: These values can be transformed using the membership functions defined by the user, e.g. when the throughput time is analyzed and the three membership functions short, average and large are defined, the membership values to all functions are stored in the membership array;
- List: These values can also be transformed using the membership functions defined by the user. The only change is that this list can contain any number of values. For every value, the membership against all functions is calculated and the maximum value for every feature is stored. For example, if the operating time of the activity *Send on* is analyzed, where *Send on* can occur multiple times within one case, the membership to all fuzzy sets, of that feature, is calculated for every operating time of that activity. For every case, the maximum value for every feature is stored.

The three scenarios, explained above, are visualized next. Table 5 defines the membership functions for the features introduced in Table 3 (*throughput time* and *operating time of "Send on"*), e.g. a throughput time between 0 and 20 days is considered as short, and a throughput time between 20 and 40 days is partly short and partly average. These membership functions are visualized in Figure 10, and can be used to create the membership array, shown in Table 6. The feature *"Send on" performed* is either True or False, and is, therefore, depicted as 0 (False) or 1 (True). The other membership values are calculated using the membership functions, e.g. a throughput time of 30 results in a membership of 0.5 for a *"Short TT"*, using equation (33):

$$f(30; 0,0,20,40) = \max\left(min\left(\frac{30-0}{0-0}, 1, \frac{40-30}{40-20}\right), 0\right) = 0.5$$

Table 5: Example linguistic labels for conversion (TT = throughput time, OT = operating time)

Feature	*Label*	*Point a*	*Point b*	*Point c*	*Point d*
Throughput	Short TT	0	0	20	40
time case	Average TT	20	40	60	70
(in days)	Large TT	60	70	100	100
Operating time	Short OT	0	0	20	40
"Send on"	Average OT	20	40	60	70
(in days)	Large OT	60	70	100	100

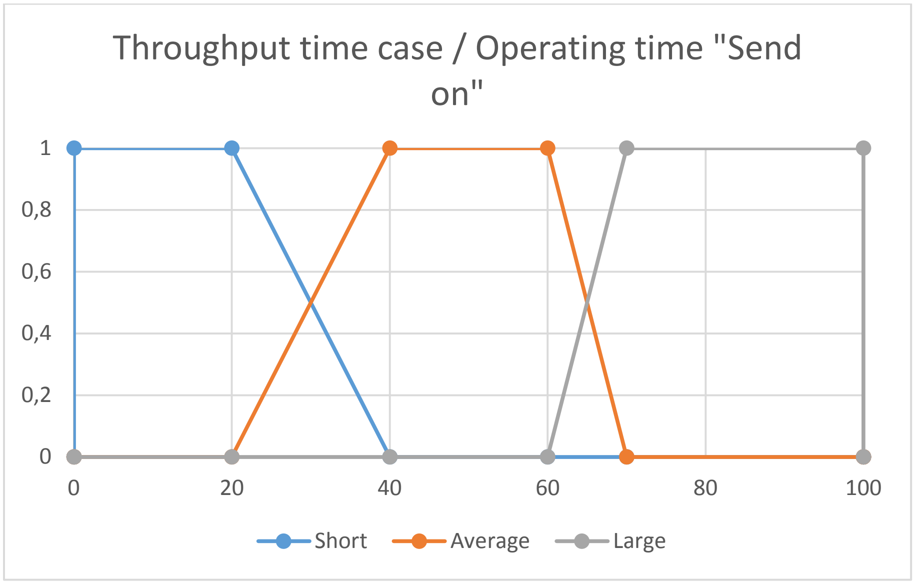


Figure 10: Example membership functions Throughput time case and Operating time "Send on"

Table 6: Example feature conversion (TT = throughput time, OT = operating time)

Case ID/ Feature	*"Send on" performed*	*Short TT*	*Average TT*	*Large TT*	*Short OT*	*Average OT*	*Large OT*
1	1	0	0	1	0.5	1	0
2	0	0.5	0.5	0	0	0	0
3	0	1	0	0	0	0	0
4	1	0	1	0	0	1	0
…	…	…	…	…	…	…	…

In addition, the combinations of features can be taken into account in this phase. To store the combinations, the algorithm loops over the data and stores the membership of every combination. Consider, for example, the combination between activities and resources, to find out which resource performed what activity. Every unique combination can be seen as a separate feature, e.g. when there are three activities (Activity A, Activity B and Activity C) and two resources (Person A, Person B), the resulting set of features is: <|Activity A||Person A|>, <|Activity A||Person B|>, <|Activity B||Person A|>, <|Activity B||Person B|>, <|Activity C||Person A|>, and <|Activity C||Person B|>, where <|Activity A||Person A|> means that Person A performed Activity A. For every case it can be stored whether these combinations occurred. As discussed in Section 5.4, no unlimited attributes are considered. If unlimited attributes are taken into account, the fuzzy sets can be considered as values for one feature (e.g. low costs, average costs or high costs). The minimal membership value of the features can be set as membership of the combination, e.g. if the costs are low with a membership of 0.7 and Person A performed that activity, the membership to the combination <low costs|| Person A|> is 0.7.

6.4 Relevant parameters for linguistic summarization

Depending on what the user wants to evaluate, different aspects need to be analyzed. In Chapter 3 and 5 various protoforms and features are discussed that can be used for different purposes, respectively. If everything is analyzed, both the running time and the amount of results will be gigantic. Therefore, a smart selection has to be made in what analysis to perform. Since the selection of the protoforms and features highly depends on both the process and what the user wants to evaluate, these have to be selected manually. Next to the selection of protoforms and features, more parameters exist that have an influence on the analysis. This section focuses on relevant parameters that can be useful for the analysis, namely:

- Quantifiers of linguistic summaries;
- Minimal truth value;
- Maximum number of summarizers;
- Maximum number of qualifiers;
- Degree of focus.

These parameters are described below. Additional parameters, that may be relevant for the analysis of sequences, are discussed in Section 7.1.

Quantifiers of linguistic summaries

All linguistic summaries contain a quantifier, e.g. *"In **most** cases there was a large throughput time"* or *"In **almost all** cases there was Person A involved"*. As discussed in Section 2.1, only relative monotonically non-decreasing quantifiers [30] such as many, most or almost all are used. The quantifiers can be seen as fuzzy sets and are defined by use of trapezoidal membership functions, as explained in Section 6.2. An example setting of the quantifiers is visualized in Figure 11.

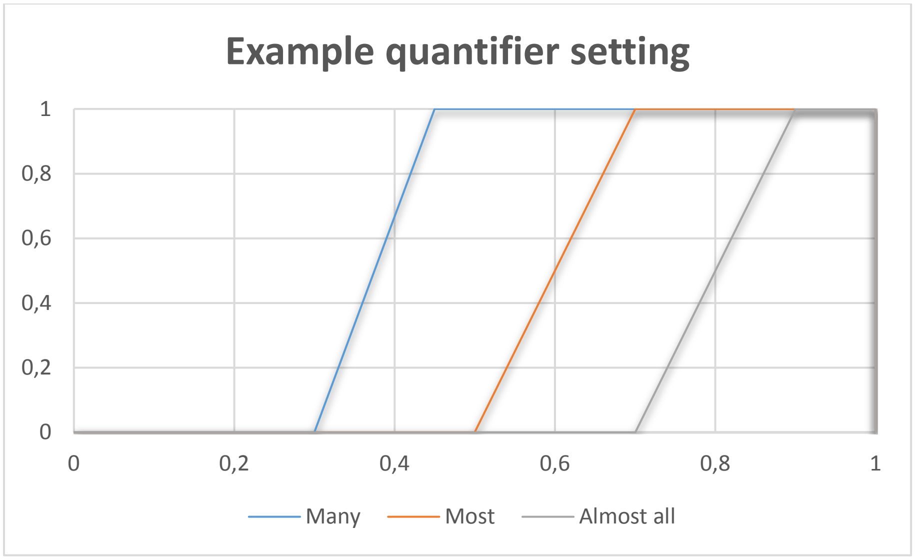


Figure 11: Example quantifier setting

Minimal truth value

Since fuzzy sets are used to define the quantifiers, one can calculate the membership of a value to one or multiple sets of the same feature. Consider the case that the quantifiers introduced in Figure 11 are used and the sentence *"In Q cases there was Sent advice"* is validated, where *Sent advice* is performed in 60 % (0.60) of the cases. This scenario is visualized in Figure 12. If Q is *Many*, the truth value of the sentence is 1; if Q is *Most*, the truth value is 0.6; and if Q is *Almost all*, the truth value is 0. A minimal truth value is introduced to determine if a sentence is valid. If this minimal truth value is set to 0.7, only the sentence with quantifier *Many* is true. However, if the minimal truth value is set to 0.5, both the sentence with quantifier *Many* and the sentence with quantifier *Most* are true. The first scenario is shown in Figure 12, where the minimal truth value is shown as the yellow dotted line and the blue dotted line represents a fuzzy proportion of P elements (0.60 in the case explained above). A certain quantifier is marked as valid for a certain feature if the membership function crosses the value (the black dotted line) above the minimal truth value (the yellow dotted line).

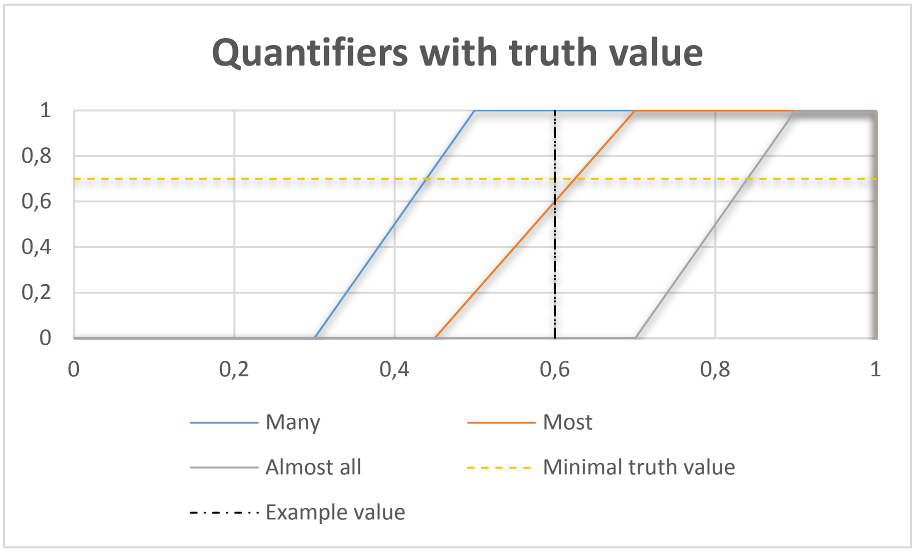


Figure 12: Quantifiers in combination with a truth value

Maximum number of summarizers

The summarizer of a sentence can focus on multiple features of a dataset, e.g. the sentence *"In most cases, there is a large throughput time and Person A is involved"* is about both the throughput time and the involvement of Person A. If too many features are added to the summarizer, the sentence might still have a large truth value, but is not understandable for the human mind. By setting a maximum number of features, sentences that contain too many features for the summarizer can be filtered.

Maximum number of qualifiers
A qualifier of a sentence can also be about multiple features of a dataset. The condition of the sentence can be made more specific by adding more features to the qualifier. This might be very interesting if one is, for example, looking why there is a *large throughput time*, e.g. *"In almost all cases, when there were many resources present and Person A performed task X, there was a large throughput time"* can give indications why the throughput time is large. However, similar to the summarizers, when too many features are added to the qualifier, the sentence might have a large truth value, but is not understandable for the human mind. Therefore, one can choose to set a maximal number of features that are contained in the qualifiers.

Degree of focus
This parameter is also related to the qualifiers. Some sentences might be valid, but are still not preferred by the user. Imagine the case that *Person A* performed *Create dossier* once, because *Person B* was ill that day. Since *Person A* was not trained for the task *Create dossier*, the operating time was larger than normal. A sentence *"In almost all cases, where Person A performed Create dossier, the operating time was large"* is true for this case, but might be unwanted, since it can be considered as an exception. The degree of focus can be set to filter out such cases, by setting a minimal threshold of the part of the cases that must satisfy the condition.

6.5 Usability of features related to the complete case
Using the logic described in this chapter, a membership array can be created that focuses on features related to the complete case. This membership array is needed as input for the generation of sentences, described in Chapter 8, and is used to generate sentences like "*In most cases, there is a large throughput time*", "*In almost all cases, when Person A performed Send on, there was a large throughput time*", or "*In almost all cases, when there was a short throughput time, there was an average throughput time for Send on*". In addition, some parameters are introduced, that can be used to scope the analysis, such that only relevant aspects are analyzed, e.g. discard a sentence if the condition is to specific or if it contains to many features for the summarizer. Since only relevant aspects are analyzed, the running time is decreased. However, the control-flow perspective of an event log can also be considered, by looking at sequences that occur, or at certain aspects of such a sequence. This is investigated in next chapter.

7 Implementation issues concerning sequences

In the previous chapter, it is discussed how features related to the complete case are handled. One may also be interested in the analysis of sequences. However, other features are relevant if sequences are analyzed. In addition, event logs can contain many sequences. If all sequences are analyzed, both the running time and the resulting set of sentences can be gigantic. In this chapter, it is discussed how sequences are handled in this thesis. Section 7.1 discusses some parameters that can be used to select only relevant sequences. After the relevant sequences are selected, there may still be many sequences left. Section 7.2 discusses how these sequences can be grouped together, such that sentences can be made focusing on a group of sequences. Section 7.3 discusses how the case focused protoforms related to the containment of sequences are handled, to get a high level overview of paths that are taken, or to determine the influence of certain sequences to other features. In addition, one can investigate in certain aspects of a sequence, e.g. the throughput time of a sequence, which is discussed in Section 7.4. Finally, Section 7.5 concludes what can be achieved with the logic described in the chapter.

7.1 Parameters for sequences

This section discusses the parameters that can be relevant when sequences are analyzed, namely:

- Maximum length of a sequence;
- Threshold sequence occurrences;
- Comparison method.

Next, the parameters are described in more detail.

Maximum length of a sequence

The sentence *"Most cases contain a sequence like <Send advice, Create dossier, Process decision>"* is very easy to understand and can therefore be useful for the user, to get a high level overview of the process flow. However, when analyzing longer sequences, the length of the sentences increases and they may be harder to understand, and are, therefore, less useful for the user. There may be cycles in the process, which can result in sentences like *"Most cases contain a sequence like <Send advice, Create dossier, Send advice, Create dossier,…>"*. By use of a maximum length of a sequence, the user is able to select what may be worth to analyze.

Threshold sequence occurrences

Not all paths are taken frequently. Some cases may go wrong and result in exceptional paths that are taken. Since one probably does not want to analyze every exceptional case, a threshold can be set for the frequency of a sequence. In this way, sequences that only occurred a few times can be discarded. This parameter is not equal to the degree of focus, since sequences are clustered, and sentences are created about the complete cluster. This is further explained in the next section.

Comparison method

Since there may be many relevant sequences that are worth analyzing, one may be interested in the clustering of sequences. In this thesis, sequences are clustered based on similarity. The similarity between sequences is measured using the Levenshtein distance [49], also known as the string-edit distance. This distance is equal to the minimal number of character edits needed to go from one string to the other one, e.g. the Levenshtein distance between *"ABC"* and *"ABD"* is 1, Since they are equal when *"D"* is replaced by *"C"*. The similarity between two strings is equal to:

$$sim_{x,y} = 1 - \frac{Levenshtein(x,y)}{(\max(length(x), length(y))} \tag{34}$$

Where Levenshtein(x, y) is the Levenshtein distance between x and y and length(x) is the number of characters of x.

The relevance of a sequence can be measured by calculating the average membership to all cases. A sequence can be added to the set of relevant sequences, if enough cases contain similar sequences. Consider, for example, the sequence *"ABC"* and the case *"ABDCE"*. The membership of this sequence to the case "*ABDCE*" can be determined by calculating the maximal similarity to all subsequences of that case, illustrated in Table 7. To get the membership of sequence *"ABC"* to the case *"ABDCE"*, the similarity of *"ABC"* already needs to be checked against ten sub-sequences. When cases get longer, the amount of comparisons that need to be made grows exponentially and, therefore, also the running time.

Table 7: Similarity between sequence "ABC" and case "ABDCE"

Subsequence of "ABDCE"	*Similarity to "ABC" (using (34))*
AB	*2/3*
BD	*1/3*
DC	*1/3*
CE	*0*
ABD	*2/3*
BDC	*1/3*
DCE	*0*
ABDC	*3/4*
BDCE	*1/4*
ABDCE	*2/5*

To deal with the problem mentioned above, one can choose to trade quality for running time. Four methods are introduced that can be chosen for measuring the membership to all cases:

- Full search;
- Double length;
- Maximal own length;
- Own length.

These methods are discussed next in more detail.

When the *full search* method is used, the corresponding sequence is compared with all other sequences that can return a higher similarity. Consider two sequences: sequence1 and sequence2. When calculating the membership of *sequence1*, *sequence2* only has to be considered when:

$$length(sequence2) \leq length(sequence1)^2 - 1 \tag{35}$$

When the *double length* method is used, *sequence1* is only compared to *sequence2* if:

$$length(sequence2) \leq length(sequence1) * 2 \tag{36}$$

Using this method, most relevant sequences are validated. Note that only exceptional cases can give a higher membership value, if their length is longer. If this is the case, the membership value does not change that much and, therefore, will probably not have a big influence on the result. For example, when checking the similarity of the sequence "*ABC*" to the case "*ADDBDDC*", the *Double length* method results in a membership of 1/3 (e.g. "*ABC*" vs "*ADD*"), and the *Full search* method results in a membership of 3/7 (compare "*ABC*" with complete case). Note that this is an exceptional case, and still the membership does not change that much.

One can also choose to use the method *maximal own length*. For this method, *sequence1* is only compared to *sequence2* if:

$$length(sequence2) \leq length(sequence1) \tag{37}$$

This method can result in different sequences that are deemed relevant, since no sequences are validated that are longer than the sequence in consideration. However, when one wants to get a high level overview of the relevant sequences, this method can be sufficient.

The last method, *own length,* is to compare sequences only to sequences of their own length. For this method, *sequence1* is only compared to *sequence2* if:

$$length(sequence2) = length(sequence1) \tag{38}$$

This method is useful if one does only want to consider sequences of the same length. However, if the sequence is longer than a certain case, it is not possible to check the sequence against its own length. In this case, the case is extended with some 'slack' activities that did not occur in the log, such that the similarity can be calculated.

7.2 Cluster sequences

This section focuses on the clustering of sequences, such that sentences can be created about the complete cluster. Clusters can be generated using several methods. The clustering of similar sequences is based on the Levenshtein distance, similar as in [24]. First, the relevant sequences are obtained, which are used to build the clustering on. As a final step, a representation is chosen for every cluster. These steps are elaborated next.

1) Get relevant sequences

In Section 7.1, three parameters are introduced that can be used for selecting relevant sequences. The first parameter (the length of a sequence) and second parameter (the occurrence of a sequence), can easily be calculated per sequence, and are used to filter sequences that exceed the length or have a too low occurrence, respectively. The third parameter (the comparison method) is used as a scope, to get the average membership of a sequence to a case. The occurrence of a sequence is crisp (i.e. it is either present or not), and the membership of a sequence is fuzzy, e.g. sequence *"ABC"* has got a membership of 0.75 to the case *"ABDCE"*, as shown in Table 7.

To make the calculation of the average membership of every sequence to all cases as efficient as possible, variables are stored and used in an efficient manner. Three lists are created for this purpose:

- The first list contains a subset of all sequences of the event log. This list is filtered on the length of a sequence, where the maximal length is set to the maximal length which can be relevant for the comparison, e.g. $maximal\ length\ of\ a\ sequence^2 - 1$ for the method 'full search' or $maximal\ length\ of\ a\ sequence * 2$ for the method 'double length';
- Another list is stored that contains all unique traces of complete cases, together with the information of the number of cases that are built of this trace;
- The last list is used to store what sequence is contained in which trace.

After the lists are created, membership values can be calculated using Algorithm 1. First the membership of all sequences to all other sequences, to which they need to be compared to, is calculated. Whether a sequence needs to be compared, depends on the length of the sequence, and the comparison method chosen, described in previous section. The maximal membership is stored per trace, since all cases with the same trace can be treated similar. If one chooses to only compare against the own length of the sequence, the similarity to traces that are shorter than the sequence is not calculated, since the trace contains no relevant subsequences. As output of this algorithm, the average membership to all cases is returned. A sequence is kept if a quantifier returns a membership greater than the minimal truth value for the average membership.

Consider the sequence *"ABCDE"* and the case *"FBGHI"*. The similarity between the sequence and case is equal to 0.2. However, B is the only activity in common. For this reason, membership values lower or equal to 0.2 are considered as noise and are removed from the analysis. The remaining values are normalized again (line 23). The chosen value is selected by performing experiments, and can be adjusted. Nevertheless, experiments show that more relevant sequence are obtained using this method.

Algorithm 1 Calculating membership of sequence to all cases

```
1:  parameters:
2:  seq is sequence that is validated
3:  sequences is the list of relevant sequences
4:  uniqueTraces is list of unique traces and their occurrence
5:  seqTrace is a list to store what sequence is contained in what trace
6:  nrOfCases is the number of cases in the event log
7:  comparisonMethod is the method used to determine whether a sequence needs to be compared
8:
9:  traceMembership = {}
10: for sequence in sequences do
11:    if needToBeCompared(sequence, seq, comparisonMethod) then
12:       similarity = Levenstein(sequence, seq)
13:       for trace in seqTrace[sequence] do
14:          traceMembership[trace] = max(traceMembership[trace], similarity)
15:       end for
16:    end if
17: end for
18:
19: sumMem = 0
20: for trace in uniqueTraces do
21:    if traceMembership[trace] is not calculated then
22:       similarity = Levenstein(trace, seq)
23:    end if
24:    sumMem += trapezoidal(traceMembership[trace], 0.2, 1, 1, 1) * uniqueTraces[trace]
25: return sumMem / nrOfCases
```

2) Create a clustering

After all relevant sequences are extracted, the sequences can be clustered, using different methods. In this thesis, the distance matrix is used, where the distance between every sequence is stored. The distance matrix is referred to as *D* and is calculated as:

$$D_{x,y} = \frac{Levenshtein(seq_x, seq_y)}{(\max(length(seq_x), length(seq_y))} \tag{39}$$

Where *Levenshtein(x,y)* is the Levenshtein distance between sequence *x* and *y*. This distance matrix can be used to create a hierarchal clustering, using the linkage method of the scipy package [50], where different methods can be used to create a different clustering, that are all based on different algorithms. This hierarchal clustering can be used to form flat clusters, using the fcluster method from the scipy package [51].

To validate the created clusters, the Correlation Cluster Validity (CCV) indices based on Pearson and Spearman correlations are used [52]. This method is chosen since it is able to validate the clustering results fully automatic. Also it is shown to produce similar results to other methods [24]. Other methods that can be considered are, for example, a visual based method based on VAT or iVAT [53], [54], or other methods, e.g. Davies-Bouldin [55] or Xie-Beni [56].

As part of the CCV indices, the dissimilarity matrix is calculated from the partition matrix. The formula for the dissimilarity matrix is stated as follows:

$$D(U) = [1]_n - \left[\frac{U^T U}{max_{i,j}(U^T U)_{i,j}}\right] \tag{40}$$

In this formula is $[1]_n$ a $n\,x\,n$ matrix that contains only 1's, where *n* is the number of sequences present in the event log, and *U* the partition matrix. In this thesis, the Pearson Correlation Cluster Validity (CCVP) Index is used. Another approach is to use Spearman's (rho) Correlation Cluster Validity (CCVS) Index. The program is built such that additional methods can be implemented easily.

Both methods compare the dissimilarity matrix (D(U)) to the distance matrix (*D)*. The CCVP method is based on the linear relationship between these matrices, where the CCVS method uses the monotonic relation between the relative ranks. The CCVS method is less sensitive to outliers and is, therefore, more robust than CCVP. Both methods return a value between -1 and 1, where the results is -1 if the dissimilarity matrix and distance matrix are not correlated, and 1 if the dissimilarity matrix and distance matrix are correlated. The methods can be calculated as follows:

$$v_{ccvp}\big(D, D(U)\big) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} A_{i,j}B_{i,j}}{\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} A_{i,j}^2}\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} B_{i,j}^2}} \tag{41}$$

$$v_{ccvs}\big(D, D(U)\big) = \left[1 - \frac{6}{n^3 - n}\sum_{i=1}^{n}\sum_{j=1}^{n}\big(D_{i,j} - D(U)_{i,j}\big)^2\right] \tag{42}$$

Where $A_{i,j} = \big(D_{i,j} - \overline{D}_{i,j}\big)$, $B_{i,j} = \big(D(U)_{i,j} - \overline{D}(U)_{i,j}\big)$, $\overline{D}$ is a matrix in which every entry is the average value of D and $\overline{D}(U)$ is a matrix in which every entry is the average value of $D(U)$.

Both the needed amount of clusters and the best clustering method highly depend on the process and are, therefore, not known by the algorithm. Therefore, the clustering is made for any relevant number of clusters and for any method. The minimal number of clusters is 1 (where all sequences are within one cluster) and the maximal number of clusters is equal to the number of sequences (where every cluster contains only one sequence). Since the CCV method is able to validate the created clusters automatically, the best combination of the number of clusters and the clustering method can be selected.

3) Select representation for the chosen clusters

After the correct clustering is created, a representation has to be selected for every cluster. The medoid sequence is chosen as the representation, since this sequence is, on average, closest to all other sequences [24]. The distance to all other sequences within the same cluster can be calculated using the distance matrix D, calculated in Step 2). For some clusters, multiple sequence can serve as medoid. In such case the shortest sequence is chosen for readability of the linguistic summaries.

7.3 Contain sequence like ABC

Protoform (8) - (11) are about the containment of sequences. A new membership array is created to be able to create sentences, related to this topic. This membership array is a $n\ x\ m$ array, where n is the number of cases and m is the number of clusters. In cell *i,j*, the membership of cluster *j,* to case *i,* is stored. The membership of a cluster within a case can be calculated using different methods. Since the medoid is closest to all relevant sequences within one cluster, it can be used to perform calculations. In this way the clustering is taken into account indirectly, since the medoid uses the sequences within that cluster. Another approach would be to use the clustering results directly and perform the calculations on all sequences within one cluster. In this case, the medoid only serves as the representation.

For both approaches, one can choose to define the membership either fuzzy or crisp. The methods are illustrated in the following example:

Consider cluster "*ABC*" that consists of the sequences "*AC*", "*ABC*" and "*ACBD*" and a case that consist of the trace "*ACBD*".

- When calculations are made using **the medoid** and a **crisp** approach is used, the membership to the case is 0, since "*ABC*" is not contained;
- When calculations are made using **all sequences** and a **crisp** approach is used, the membership to the case is 1, since "*AC*" is contained;
- When calculations are made using **the medoid** and a **fuzzy** approach is used, the membership to the case is 2/3, since this is the similarity between sequence "*ABC*" and "*AC*";
- When calculations are made using **all sequences** and a **fuzzy** approach is used, the membership to the case is again 1, since "*AC*" is contained.

Note that, if the crisp method returns a membership of 1, the fuzzy method will also return 1. However, if the crisp method returns a membership of 0, the fuzzy method returns a value in-between 0 and 1.

When calculations are made using the medoid, sentences may be easier to understand, since one does not need to know all sequences that are contained within a cluster. However, if the medoid is chosen, the clustering results are taken into account indirectly. When using all sequences in combination with the fuzzy method, the sentences can become (almost) untraceable (i.e. it is not known where the sentence is based on). In this thesis, it is chosen to perform the calculations for all sequences in combination with a crisp approach. This method is chosen, since the clustering is shown to the user, and the user is able to choose other clustering methods. Since the user has to decide on the clustering, it may be expected that those results are taken into account directly. In addition, it is easier to trace the sentences, features can be calculated more easily, since the fuzzy aspect does not need to be taken into account for every feature, which has got a great influence on the running time.

7.4 Sequence like ABC was Ps

The last protoforms that are in the scope of this thesis are the sequence focused protoforms ((12) - (15)). For these protoforms a different set of features is relevant. Sentences like *"In most cases, the activity Process decision was performed for a sequence like <Process decision, Create dossier>"* provide non-trivial information and, therefore, they do not have to be analyzed. A new set of features is introduced, that are relevant for sequence focused protoforms. This set is a subset of the features introduced in Table 2 and is shown in Table 8. Column 2 is based on the same values, discussed in Section 6.1. The last column shows an explanation for the given feature.

The throughput time, waiting time and operating time must be normalized based on the length of a sequence, since sentences are based on the complete cluster. Consider, for example, the cluster <Process decision, Create dossier>. If the sequence <Process decision, Create dossier, Archive> is also contained in this cluster, the throughput time, waiting time and operating time are possibly larger for this sequence, since more activities are executed. To mitigate this effect, the times are normalized based on the length of the sequence.

For sequence focused protoforms no case attributes are analyzed, since they are equal for the complete case. A sentence like *"In most cases, that contain a sequence like <Process decision, Create dossier>, there was Request building permit for 'Reason'"* gives the same information as *"In most cases, there was Request building permit for 'Reason' for a sequence like <Process decision, Create dossier>"*.

In this thesis, it is chosen to base the feature '*number of distinct values selected for sequences*' and the feature *'number of times a value is selected for sequences'* on their percentage based versions, introduced in Table 4. As discussed in Section 5.3, membership functions need to be constructed for every sequence if the frequency based version is used, since this version is length specific. The percentage based version adapts to the length and needs, therefore, less user interaction, while providing similar results.

Table 8: features for sequence focused protoforms

Feature	How to store	Explanation
Throughput time sequence	List	Throughput time of every sequence of the cluster, that is present in the case, is stored and normalized on length.
Waiting time sequence	List	Waiting time of every sequence of the cluster, that is present in the case, is stored and normalized on length.
Operating time sequence	List	Operating time of every sequence of the cluster, that is present in the case, is stored and normalized on length.
Value selected – unlimited activity attribute	List	Multiple values for the activity attribute can be logged for the same case.
Value selected – limited activity attribute	Boolean	A certain value was logged or not.
Number of distinct values selected for sequences	List	Different number of distinct values can be logged for the different sequences in the cluster.
Number of times a value is selected for sequences	List	Different values can be logged for the different sequences in the cluster.

7.5 Usability of features related to sequences

This chapter describes how sequences are analyzed in this thesis. Using the logic described in this chapter, the membership of features related to sequences can be calculated, and the membership arrays for the containment of sequences and the membership array for sequence focused protoforms can be created. These membership arrays can be used in the generation of sentences, described in the following chapter, and are used to generate sentences like "*Most cases contain a sequence like <Create dossier, Process decision>*" or "*Most cases contain, that contain a sequence like <Register appeal, Confirm reception>, contain a sequence like <Create dossier, Process decision>*" for the membership array related to the containment of sequences, and "*In most cases, there was a large throughput time for a sequence like <Create dossier, Process decision>*" or "*In most cases, when there was a large throughput time for a sequence like <Create dossier, Process decision>, there was Person A involved for a sequence like <Register appeal, Confirm reception>*" for the membership array related to the sequence focused protoforms. In addition, the membership arrays created in this chapter can be used in combination with the membership array that focuses on features related to the complete case, discussed in Chapter 6, to generate sentences like "*Most cases, where Person A performed Register appeal, contain a sequence like <Create dossier, Process decision>*" or "*In almost all cases, when there was a large throughput time for a sequence like <Create dossier, Process decision>, there was a large waiting time*".

8 Generating linguistic summaries

In previous chapters, it is discussed what features may be relevant for the analysis of processes, what parameters can be set, and how the data can be converted into a format that can be used in the algorithm created. As a final part of the program, linguistic summaries are created focusing on the process. The input for this part of the algorithm, is one membership array for the summarizers, one membership array for the qualifiers, and parameters (e.g. the degree of focus) that need to be taken into account. In this chapter, it is discussed how all protoforms, that are in the scope of this thesis, are handled. Section 8.1 describes how sentences are created and Section 8.2 elaborates on the pruning of these sentences, to discard sentences that are not relevant for the user.

8.1 Sentences generated by the algorithm

Table 9 shows what input is needed for what protoform. The membership arrays are either based on the membership array for the features related to the complete case (Section 6.3), the membership array for the containment of sequences (Section 7.3), or the membership array for sequence focused protoforms (Section 7.4). In column two of Table 9, the section numbers are used to indicate which membership array is used for the summarizers and which membership function is used for the qualifiers. Note that the membership array for the summarizers can be equal to the membership array for the qualifiers. However, the features of the summarizer cannot be used in the qualifier and, vice versa, the features of the qualifiers cannot be used in the summarizer. This is done to avoid sentences like *"In most cases, where there was a large throughput time, there was a large throughput time"*.

In Section 6.4 and 7.1, parameters are introduced that can be relevant for the analysis of processes. Different parameters are needed for every protoform, e.g. the degree of focus is not needed when no qualifier is used. An overview of the parameters needed for every protoform is shown in Appendix D. For the protoforms related to sequences, the maximum number of summarizers and/or the maximum number of qualifiers is set to one. This is done for the simplicity of the sentences.

Table 9: Input per protoform

Protoform	Input	Comment
Protoform (6): In Q cases, there was P	*P*: 6.3 *R*: -	
Protoform (7): In Q cases, when condition R was fulfilled, there was P	*P*: 6.3 *R*: 6.3	Features in the summarizer are not contained in the qualifier for the same sentence
Protoform (8): Q cases contain a sequence like ABC	*P*: 7.3 *R*: -	No more than one summarizer
Protoform (9): Q cases, when condition R was fulfilled, contain a sequence like ABC	*P*: 7.3 *R*: 6.3	No more than one summarizer
Protoform (10): In Q cases, that contain a sequence like ABC, there was P	*P*: 6.3 *R*: 7.3	No more than one qualifier

Protoform	Input	Comment
Protoform (11): Q cases, that contain a sequence like ABC, contain sequence like XYZ	*P*: 7.3 *R*: 7.3	Features in the summarizer are not contained in the qualifier for the same sentence No more than one summarizer/ qualifier
Protoform (12): In Q cases a sequence like ABC was P_S	*P*: 7.4 *R*: -	No more than one summarizer
Protoform (13): In Q cases, when condition R was fulfilled, a sequence like ABC was P_S	*P*: 7.4 *R*: 6.3	No more than one summarizer
Protoform (14): In Q cases, when a sequence like ABC was R_S, there was P	*P*: 6.3 *R*: 7.4	No more than one qualifier
Protoform (15): In Q cases, when a sequence like ABC was R_S, a sequence like XYZ was P_S	*P*: 7.4 *R*: 7.4	Features in the summarizer are not contained in the qualifier for the same sentence No more than one summarizer/ qualifier

The generation of the sentences is based on [57] and is explained in three algorithms, explained next: Algorithm 2, Algorithm 3 and Algorithm 4,. Algorithm 2 is the main method. If a qualifier is needed (depending on the protoform), all possible qualifiers are calculated using Algorithm 3. For every possible qualifier, the membership array, used to find the summarizers, is converted such that it conforms to the qualifiers set. Valid summaries are detected, using Algorithm 4. After creating all sentences, the resulting set can be pruned, e.g. the sentence *"In most cases, there is a large throughput time"* is a subset of the sentence *"In most cases, there is a large throughput time and Person A is involved"* and can, therefore, be pruned.

Example: One wants to analyze the appeal dataset, and focuses on analyzing protoform (7). Chapter 6 describes how the membership array is constructed for this protoform (the membership array for the summarizer and qualifier are equal for this protoform). For this protoform, a qualifier is needed, and, therefore, first all qualifiers are determined using Algorithm 3. In this algorithm, it is determined which features can be used as a qualifier, based on the degree of focus. If the degree of focus of, for instance, the feature '*a short throughput time*' is not high enough, the feature '*a short throughput time **and** Person A was involved*', does not have to be validated, since the degree of focus can only decrease if new features are added. After all qualifiers are determined, the summarizers can be determined. First, the membership array is converted such that it conforms to the qualifier set, e.g. if the qualifier is '*high throughput time*', and the membership of a certain case to this feature is 0.5, the membership all features is set to a maximum of 0.5 for that case. After the array is converted, it can be used to find the summarizers, using Algorithm 4. In this algorithm, every relevant (set of) feature(s) is validated, that can return a valid statement, e.g. if no quantifier is true for the feature '*low waiting time*', the feature '*low waiting time **and** Person A was involved*' does not have to be validated, since the truth value can never increase by adding features. Finally, if all summarizers are found, they can be pruned, e.g. if both the summarizer '*large throughput time*' and '*large throughput* *time* ***and*** *Person A is involved*' are valid for the same quantifier (e.g. '*most*') and qualifier (e.g. '*large waiting time*'), the first summarizer can be discarded, since it does not provide any additional information. The summarizers left can be used in the generation of the sentences, by filling the protoform, e.g. if the summarizer '*large throughput time*' is valid for the quantifier '*most*' and qualifier '*large waiting time*', the sentence "*In most cases, when there was a large waiting time, there was a large throughput time*" is generated.

Algorithm 2 Generate sentences

```
parameters:
memP is the membership array for the summarizers
memR is the membership array for the qualifiers
quantifiers is an ordered list of relevant quantifiers
minTruth is the minimal truth value
maxLenP is the maximum number of summarizers
maxLenR is the maximum number of qualifiers
degreeFoc is the degree of focus
nrOfCases is the number of cases in the event log
needQualifier is a Boolean shows whether a qualifier is needed
protoform is the protoform to be analyzed

sumAndQual = {}              //Used to store summarizers for given qualifier
if needQualifier do
   possR = findPossibleQualifiers(memR, maxLenR, degreeFoc)
   for qualifier in possR do
      newMemP = setR(qualifier, memP)
      sumAndQual[qualifier] = findCorrespondingP(newMemP, quantifiers, minTruth, maxLenP, nrOfCases)
   end for
else do
   sumAndQual [] = findCorrespondingP(newMemP, quantifiers, minTruth, maxLenP, nrOfCases)
end if
pruneSummarizers(sumAndQual)
sentences = createSentences(sumAndQua, protoforml)
```

The methods used in Algorithm 2 are explained below:

- **findPossibleQualifiers:** Calculate all qualifiers that might return a valid sentence. This method is further explained in *Algorithm 3: Find possible qualifiers*;
- **setR:** Convert the membership array for the summarizers such that it conforms to the qualifiers set. For example, if the condition is *'low costs and large throughput time'*, and a case has got *'low costs'* with a membership of 0.5 and a *'large throughput time'* with a membership of 1, the maximal membership of a feature cannot exceed 0.5 for that case;
- **findCorrespondingP:** Find all summarizers which return valid sentences. This method is further explained in *Algorithm 4: Find valid summarizers for one or no qualifiers*;
- **pruneSentences**: Delete superfluous sentences, which is further explained in Section 8.2;
- **createSentences:** Create sentences, by filling in the summarizer and qualifier in the corresponding protoforms.

Algorithm 3 is used to find all possible qualifiers. A qualifier is valid, if the average value of all cases to the qualifier, is higher than the degree of focus. To reduce the running time, qualifiers are pruned if they give no extra information. In addition, a qualifier should only be validated if all of its subsets return valid qualifiers. For example, if the average membership value of the qualifier *large throughput time* is only in 0.1, the qualifier *large throughput time **and** Person A is involved* can never be higher than 0.1.

Algorithm 3 Find possible qualifiers

```
parameters:
memR is the membership array for the qualifiers
maxLenR is the maximum number of qualifiers
degreeFoc is the degree of focus

possR = []
for feature in memR do
    if averageMembership(feature) > degreeFoc then
        possR.add(feature)
    end if
end for

curLength = 1
while new items are added to possR and curLength < maxLenR do
    curLength += 1
    for item in findPossibleItems(possR) do
        if averageMembership(minPerRow (item)) > degreeFoc then
            possR.add(item)
        end if
    end for
    possR = pruneR(possR)
end while
return possR
```

The methods used in Algorithm 3 are explained below:

- **averageMembership:** Calculate the average membership value of the feature to all cases;
- **findPossibleItems:** Find all qualifiers/summarizers that might have a high enough truth value, as explained in Section 6.4. For example, the combinations of the features *A, B and C* is only considered if *A and B*, *A and C*, and *B and C* have got a high enough degree of focus for the given qualifier, and truth value for the summarizer;
- **minPerRow:** To check the membership of a set of features to a case, the minimal membership of all features is taken. For example, if there is a *large throughput time* with a membership of 0.8 and *Person A is involved* in the case (membership of 1), the membership of the feature *large throughput time and Person A* is min(0.8, 1), which is equal to 0.8;
- **pruneR:** Delete unnecessary qualifiers. Consider that the degree of focus of the condition *Person A was involved and Person B was involved* is high enough. This condition does only have to be validated if there is an information gain. This implies that the degree of focus of the new condition can never exceed the degree of focus of its sub conditions. e.g. if *Person B*

was always involved if *Person A* was involved, all sentences that are valid for *Person A*, are valid for *Person B*. This type of pruning is performed to reduce the running time of the algorithm. The resulting set of sentences does not change using this type of pruning, since the sentences pruned are also pruned using the pruning steps discussed in Section 8.2.

Algorithm 4 is used to find all valid summarizers for a given qualifier, given a membership array of all features. Similar to the qualifiers, a set of summarizers is only validated if all of its subsets return true sentences. For example, if the sentence *"In most cases, there is a large throughput time"* is not valid, the sentence *"In most cases, there is a large throughput time **and** Person A is involved"* can never be valid. To be able to create simple and compact sentences, only related summarizers are considered. By doing so, the number of created sentences decreases, the sentences are shorter, and the sentences are easier to comprehend, without any information loss [57].

Algorithm 4 Find valid summarizers for one or no qualifiers

```
parameters:
memP is the membership array for the summarizers
quantifiers is an ordered list of relevant quantifiers
minTruth is the minimal truth value
maxLenP is the maximum number of summarizers
nrOfCases is the number of cases in the event log
qualifier is the qualifier to be checked

validP = {}                // Used to store valid summarizers
for feature in memP do
  if feature not in qualifier then
    for quantifier in quantifiers do
      if truth(feature, quantifier) > minTruth do
        validP[feature] = quantifier
      end if
    end for
  end if
end for

curLength = 1
while new items are added to validP and curLength < maxLenP do
  curLength += 1
  for item in findPossibleItems(validP) do
    for quantifier in quantifiers do
      if checkRelated(item) and truth(minPerRow(item, qualifier), quantifier) > minTruth then
        validP[feature] = quantifier
      end if
    end for
  end for
end while
return validP
```

The methods used in Algorithm 4, that are not used in Algorithm 3, are explained next:

- **truth:** Calculate the truth value of the value to all quantifiers, return the most specific quantifier that is valid. The truth value is calculated, using the statistics shown in Appendix C;

- **checkRelated:** Check whether the summarizers are correlated. The set of features *A, B and C* is correlated if *A and B*, *A and C*, and *B and C* are related, and if the truth values are high enough for all:

$$Q\ A\ y's\ are\ B\ and\ C$$
$$Q\ B\ y's\ are\ A\ and\ C$$
$$Q\ C\ y's\ are\ A\ and\ B$$

In this thesis, *Q* is set to *almost all*, which is equal to the trapezoidal function trapezoidal *(almost all, 0.7, 0.9, 1, 1)* and the truth value must be at least 0.7. These parameters are chosen since they are proven to give good results in experiments [57].

8.2 Pruning of superfluous sentences

There are several pruning techniques that can be applied to filter the created sentences. Consider for example the sentences *"In most cases, there is a large throughput time"* and *"In most cases, there is a large throughput time and Person A was involved"*. The first sentence is contained in the second sentence and can, therefore, be pruned. In this section, three types of pruning are discussed:

- Pruning based on quantifiers;
- Pruning based on features;
- Pruning based on implication.

The first two types of pruning are based on [24] and [57], and can reduce the resulting summaries by 80%-100%. The last pruning type is not implemented due to timing reasons and is left for future research.

Pruning based on quantifiers

The first type of pruning is based on the quantifiers. If multiple quantifiers are validated as true, only the most specific one is taken. Consider, for example, the sentence *"In almost all cases, there was Register appeal"*. If this sentence is true, similar sentences with a less specific qualifier, such as *"In most cases, there was Register appeal"* and *"In many cases, there was Register appeal"* are true. However, if the sentence *"In many cases, there was Sent on"* is true, this does not imply that more specific quantifiers result in valid sentences. Note that this type of pruning can be applied only if relative monotonically non-decreasing quantifiers [30], such as most or almost all, are used. This type of pruning can be expressed as: discard $Q_1\ R\ y's\ are\ P$, if there exists a summary $Q_2\ R\ y's\ are\ P$, such that $Q_2 \subseteq Q_1 (Q_1\ is\ less\ specific\ than\ Q_2)$.

Pruning based on features

This type of pruning can be applied to remove a sequence with an equal or more specific condition, and an equal or less specific summarizer, e.g. the sentence *"In most cases, when Person A was involved, there was a large throughput time"* can be discarded if there exists a sentence *"In most cases, there was a large throughput time and a large waiting time"*. This type of pruning can be expressed as: discard $Q\ R_1\ y's\ are\ P_1$, if there exists a summary $Q\ R_2\ y's\ are\ P_2$, such that $R_2 \subseteq R_1$ and $P_1 \subseteq P_2$ [57]. The condition (R) can be an empty set.

Pruning based on implication
The last type of pruning is based on what is represented by the sentence which can be useful for both the combinations of features and the protoforms related to sequences. The second statement logically implies the first, so it's enough to state the second. Consider, for example, the sentences *"In most cases, when Create dossier was performed by Person A, there was a large throughput time"* and the sentence *"In almost all cases, when there was Create dossier, there was a large throughput time"*. From the first sentence, it seems that *Person A* has something to do with the *large throughput time*, but the second sentence proves that this is not the case. Since *Person A performed Create dossier* is seen as one single feature, this sentence is not pruned in the pruning step based on the features.

Protoforms related to sequences can also be pruned, based on the information they represent. Consider, for example, the sentences *"Most cases contain a sequence like <Create dossier, Process decision>"* and *"Most cases contain a sequence like <Send advice, Create dossier, Process decision>"*. In this case, the first sentence does not seem to give any trivial information. However, since sentences are created focusing on a group of sequences instead of on a specific sequence, this topic needs some more investigation. It occurs that the cluster name is a subset of the other cluster name, as in the sentence shown above, where not all sequences of that cluster are a subset of the other cluster, e.g. <Create dossier, Process decision> might contain the sequence <Write draft advice, Create dossier, Process decision>

It occurs that the information represented within the same sentence is irrelevant, e.g. *"In almost all cases, that contain a sequence like <Create dossier, Process decision>, there was Create dossier"*, *"In most cases where there was Person A for <Send advice, Create dossier, Process decision, Archive>, there was Person A "*, or *"In most cases where there was Person A for <Send advice, Create dossier, Process decision, Archive>, there was Person A for <Create dossier, Process decision>"* are all trivial and can, therefore, be left out.

9 Evaluation

This chapter describes the evaluation of the master thesis. Two case studies are performed for the scope of this thesis; one case study focuses on the appeal dataset and one case study focuses on audit purposes. The set-up of the evaluation is described in Section 9.1. Next, in Section 9.2, the results of the case study that focuses on the appeal dataset are given. Finally, in Section 9.3, the results of the case study, that focuses on audit purposes, is given, where the focus of this thesis is on user acceptance.

9.1 Set-up

Since the tool constructed is only a prototype, it would not be fair to compare this with a completely developed product, like Disco [58]. The goal of the evaluation is to get an understanding whether people believe that the use of linguistic summarization will improve the way that processes are analyzed. For this reason, it is chosen to evaluate the prototype by means of a demonstration in combination with a questionnaire. Both practitioners and researchers have been asked, to get an understanding of both groups of users. In total, thirteen people have been asked to fill in the questionnaire, from which six are researchers and seven are practitioners. Before the questionnaire was filled in, a short demonstration was given, to ensure one knows what to expect of such a tool; which parameters have to be set and what kind of results can be obtained. Table 10 shows an overview of the skill levels of all participants.

Table 10: Skill level of population asked

Skill / Amount of tools	*One tool*	*Multiple tools*
Beginner	0	2
Intermediate	0	4
Expert	3	4

The questionnaire is designed using the Technology Acceptance Model (TAM) [59], [60], and is constructed by combining several questionnaires [59], [60], [61], and [62]. The resulting questionnaire can be found in Appendix E. The questionnaire consists of a total of 26 questions, where the first 22 questions are based on the TAM; six questions focus on the perceived usefulness, another six questions on the perceived ease of use, four questions on the intention to use, two questions on the output quality, and four questions on the ability to demonstrate the results. The last four questions are asked to get both an understanding of the population that filled in the questionnaire and how linguistic summarization is seen in comparison with other tools.

Before the questionnaire was filled, a demonstration was given, based on the appeal dataset. The second case study, related to audit purposes, is evaluated using a semi structured interview with a process expert and focuses on the usability of such a tool for an auditor.

9.2 Results of case study on appeal dataset

In this section, the results of the case study on the appeal dataset are discussed. First, some insights are discussed obtained during the case study, including some statistics about the performance of the algorithm and a comparison to a classical process model. Section 9.2.2 discusses the results of the questionnaire.

9.2.1 Insights obtained

The event log consists of 7,604 event, and a total of 1,268 cases. The average throughput time of a case is approximately 42 weeks, while the process can even take up to two years. Therefore, it might be interesting in analyzing the throughput time, and get indications about root causes. For this dataset, the following aspects are logged:

- Activities that are executed for every case;
- Start timestamps;
- End timestamps;
- Person that performed the activity;
- Origin (e.g. a building permit) of the case are logged.

In this case study, it is shown that the algorithm, proposed in this research, produces some interesting insights, even for this rather small event log.

Appendix F shows the process model, created using Disco [58]. In this model, it can be seen that the waiting time before the activity 'Archive' is between 20 and 29 weeks. In addition, the operating time of the activities 'Register receipt of documents' (25 days), 'Result hearing / Write advice' (36 days) and 'Process decision' (24 days) is also quite large. However, root causes are hard to determine using this process model.

The algorithm, created in this research, is used to obtain new insights about the event log. Since the throughput time depends on both the waiting time and operating time, all three features are analyzed. In addition, it is analyzed which activities are performed, the number of resources that are active within one case, whether a certain resource performed many activities, the origin of the case, and the combination between resources and activities (to see what resource performed what activity). This results of 371 features that are analyzed, t related to the complete case. First, the focus is on three protoforms that consider features related to the complete case, to get some global insights on the times, namely: 1) '*In Q cases, there was P*', 2) '*In Q cases, when condition R was fulfilled, there was P*', and 3) '*In Q cases, that contain sequence like ABC, there was P*'.

Some statistics about the running time of the algorithm, and the number of created sentences are shown in Table 11. In this table, the influence of the number of summarizers, number of qualifiers and degree of focus is shown. For the last four columns, two numbers are stated. For example, for the first row represents the analysis where only one summarizer and one qualifier are chosen, and the degree of focus is set to 0.1. This analysis results in 12 sentences for protoform 1), where only one sentence focuses on the throughput, waiting, or operating time. In addition, 371 sentences are generated for protoform 2), and 51 sentences for protoform 3), where 44 and 11 sentences focus on these times, respectively. Last the running time of the complete analysis was 0.82 seconds. However, when one focuses on only the times, the running time is 0.22 seconds. It can be seen that the technique is able to generate the sentences quite fast. However, the parameters chosen have got a large impact on both the running time and the number of sentences that are created.

Table 11: Statistics appeal dataset

Number of summarizers	Number of qualifiers	Degree of focus	Number of sentences protoform 1)	Number of sentences protoform 2)	Number of sentences protoform 3)	Running time (sec)
1	1	0.1	12/1	371/44	51/11	0.82/0.22
2	1	0.1	12/1	1412/44	180/11	1.08/0.25
1	2	0.1	12/1	812/94	51/11	5.85/2.00
2	3	0.1	12/1	5651/129	180/11	19.52/4.08
2	3	0.05	12/1	18489/371	180/11	66.59/14.09
2	3	0.2	12/1	908/54	180/11	5.99/1.26

There are many sentences that might be relevant to analyze, e.g. "*In most cases, when Person A performed the activity Confirm reception, there was a large waiting time*" or "*In almost all cases, when Person A performed the activity Confirm reception, there was a large throughput time"* indicate that the throughput and waiting time are large, if *Person A* performed the activity *Confirm reception*, which is an activity at the start of the process. It can, for example, be the case that *Person A* is not trained for this task. It could also indicate that this task could have a large impact on the times when performed incorrect. On the other hand, it could be the case that *Person A* handles different kind of tasks than other persons, which take longer to complete. Therefore, these sentences can be used as indications of root causes, that have to be investigated further.

Such insights are more difficult to obtain in the process model. Another sentence that can be relevant is "*In almost all cases, when there was Person F performed many activities, there was large throughput time*", which can again have different reasons, e.g. because there is not much handing over of work, or because the tasks that *Person F* performs take longer to complete. In addition, sentences like "*In many cases, when the origin was Granted building permit, there was a large waiting time*" and "*In almost all cases, when the origin was Granted building permit, and the activity Register receipt of documents is performed, there was a large throughput time*" can indicate that cases considering a building permit take longer to complete, especially if the documents are registered.

The sentence "*In almost all cases, that contain a sequence like <Send advice, Create dossier, Process decision, Archive>, there was a large throughput time*" indicates that the throughput time is large, if the sequence <Send advice, Create dossier, Process decision, Archive> occurred. One could investigate under which conditions there is a large throughput time for this sequence, by analyzing the sequence in more depth. When this analysis is performed, sentences like "*In nearly all cases, when Person A performed the activity Confirm reception, there was a long throughput time of a sequence like <Send advice, Create dossier, Process decision, Archive>*" and "*In nearly all cases, when Person A performed the activity Register appeal, there was a long throughput time of a sequence like*

<*Send advice, Create dossier, Process decision, Archive*>” are created. These sentences indicate, that if *Person A* performed a task in the beginning of the process, the throughput time of a sequence, at the end of the process, takes a long time.

Concluding, the technique proposed already produces insights that are hard, if not undoable, to find using other techniques. Even for the rather small event log used in the case study, there are many interesting sentences, that need further investigation. However, the technique returns indications of root causes, which need to be validated with a process expert for a full understanding.

9.2.2 Results of questionnaire

In this section, the results of the questionnaire are discussed. In Table 12, the results of all questions, that are based on the TAM, are shown, which are visualized in Figure 13. The score of each question is based on Table 12, where ‘Strongly Disagree’ has got a score of 1, ‘Disagree’ of 2, etc. In addition, colors are used to indicate the height of the score, where red indicates the answer ‘Strongly disagree’, orange ‘Disagree’, yellow ‘Neutral’, light green ‘Agree’, and dark green ‘Strongly agree’, e.g. if the bar stops in the orange part, the average score lies within ‘Disagree’. Almost all participants see the usefulness of using linguistic summarization and have intentions to use it. However, since the tool needs much user input, some people doubt about the ease of use. Most participants believe that the quality of the output generated using linguistic summarization is high, but it might be hard to fully understand the system’s output, i.e. a process model is easier to understand at first sight. For example, when the sentence *“In almost all cases, where Person A performed task X, there was a large throughput time”* is generated, this could raise questions, like: *“Was the throughput time also large when other persons executed this task?”* or *“Was the throughput time significantly larger than when other persons executed this task?”*. To answer these questions, one can look at the other sentences that are created for the process, e.g. whether an equal sentence also exists for other resources. However, this could increase the running time of the algorithm drastically, especially when a lot of sentences are created. This topic is left for future research.

In addition, most participants believe that they could explain the benefits of using linguistic summarization. Note that RD4 (see Appendix E) is the only question that is asked in a negative way. Therefore the outcome ‘Disagree’, is a good result for this question. It should be noted that RD2 is only filled by twelve participants (see Table 12), because the thirteenth participant was not able to answer the question with the given context.

Table 12: Results of the questionnaire

Construct	Variable	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
Perceived Usefulness (PU)	PU1	0	0	1	11	1
	PU2	0	1	3	9	0
	PU3	0	1	0	12	0
	PU4	0	0	0	11	2
	PU5	0	0	4	6	3
	PU6	0	0	0	8	5
Perceived Ease of Use (PEOU)	PEOU1	0	2	2	6	3
	PEOU2	0	2	6	4	1
	PEOU3	0	1	5	5	2
	PEOU4	0	1	2	8	2
	PEOU5	1	0	4	8	0
	PEOU6	0	1	5	6	1
Intention To Use (ITU)	ITU1	0	0	1	8	4
	ITU2	0	0	2	8	3
	ITU3	0	1	4	5	3
	ITU4	0	0	3	8	2
Output Quality (OQ)	OQ1	0	0	4	8	1
	OQ2	1	4	1	5	2
Result Demonstrability (RD)	RD1	0	0	0	8	5
	RD2	0	1	1	8	2
	RD3	0	0	2	9	2
	RD4	2	10	0	1*	0

* One participant filled in ‘Agree’ for RD4, which is possibly due to the fact that the question is not read extensively, since other questions about the result demonstrability were answered positively.

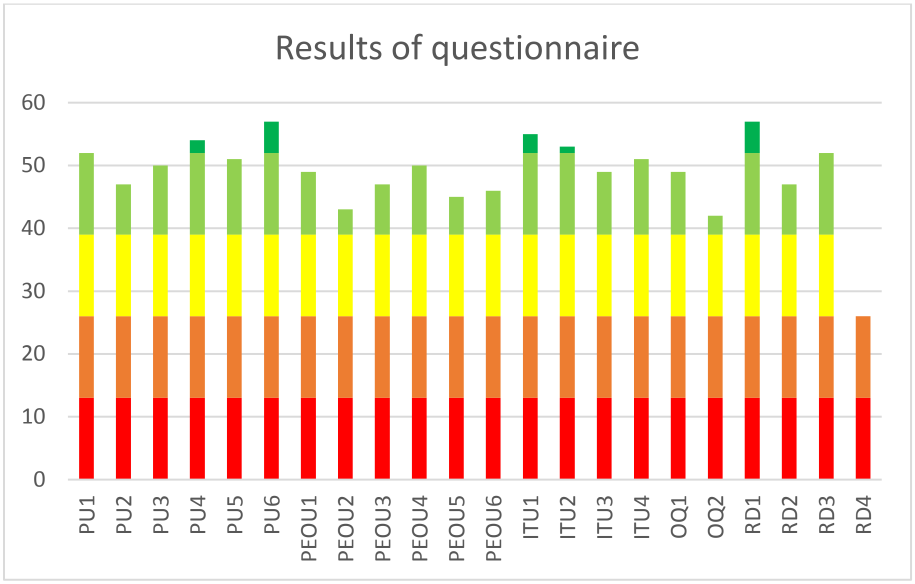


Figure 13: Aggregated results of questionnaire

As can be seen in Table 13 and Table 14, all participants believe in the use of linguistic summarization to analyze processes. One participant was really doubting the question whether linguistic summarization could be seen as an extension, or as a standalone approach, and selected both options.

Table 13: Results how the different skill levels would use linguistic summarization

Skill / Usage	*Not use Linguistic summarization*	*Use in combination with tools I currently use*	*Only use Linguistic summarization*
Beginner	0	2	0
Intermediate	0	4	0
Expert	0	7	0

Table 14: Results how the different skill levels see the results produced by linguistic summarization

Skill / See results as	*Noise*	*A subset*	*An extension*	*A standalone approach*
Beginner	0	0	2	0
Intermediate	0	0	3,5	0,5
Expert	0	0	7	0

9.3 Results of case study for audit purposes

As part of the evaluation of this thesis, a case study is performed for audit purposes. A possible check during an audit can be the evaluation of duplicate purchase invoices. It can occur that multiple payments are performed for the same invoice, for example, if someone accidentally recorded the same invoice twice with different supplier names. One can create an overview of possible duplicates, by comparing all purchase invoices on overlapping fields, such as the price. However, to determine whether the possible duplicate purchase invoice is a real duplicate, the invoices have to be compared manually, which can be a hard and time-consuming task, and it has to be performed over and over every year. Therefore, one could benefit in gaining some insight into the process, to understand the patterns that can lead to duplicate invoices.

The case study focuses on this example. It gives insight in the procurement process and tries to give indications why a duplicate purchase invoice was made. However, the case study does not investigate the results that are produced by the technique in depth, since a lot of client input is needed for this part, e.g. what can be seen as root causes and whether the membership functions are defined correctly. Therefore, the focus of this case study is to determine whether the technique, and the results produced by the technique, might be useful for audit purposes. The technique is evaluated using a semi structured interview with a process expert.

The process expert believes that the technique is, undoubtedly, useful for audit purposes. The results produced by the technique are seen as an extension to the current analysis. The technique can be used after the duplicate invoices are detected, to find any patterns in the data. Finding such patterns can be very hard and time-consuming when this has to be done manually and, therefore, the process expert believes that this technique might be very useful in the automation of this task. In addition, the results produced by the technique are more complete, since it focuses on every available combination of attributes, what can never be achieved when performed manually. The technique already produces interesting results, that might need some further investigation. However, since this highly depends on the client, this is not investigated further.

However, the technique needs a lot of user input and, therefore, the process expert discusses that the technique is best useable if it is applied on a client that is already known (such that the parameters can be set more easily) and where duplicate invoices are already analyzed before. Still, the parameters that have to be set, have to be tweaked a lot. In addition, the process expert believes that there are many improvements possible, which can make the technique even more valuable, e.g. ordering of sentences on importance, or parameters that are proposed (or set) automatically.

Concluding, the process expert believes that the results are already useable for detecting patterns in the data, that can help to find root causes of, for instance, duplicate invoices. However, the technique can still be improved, to higher the usability.

10 Conclusion

This chapter summarizes and concludes this master thesis. Section 10.1 provides the concluding remarks and answers the initial research question. Limitations of the current technique are described in Section 10.2. Next, suggestions made for future research are discussed in Section 10.3, and some recommendations can be found in the last section, Section 10.4.

10.1 Concluding remarks

In this master thesis, a new approach is investigated that helps analyzing business processes. This approach uses linguistic summarization techniques to automatically generate statements in natural language, that describe characteristics about an event log. This research contributes to the state of art, since during the literature review, several challenges are defined, which are solved in this research, e.g. event logs are investigated in general and sentences are pruned and parameters can be set for the simplicity of the generated sentences. In addition, this research investigates the generation of sentences that focus on a particular sequence, instead of the complete case, to get more detailed feedback on specific parts of the process. A prototype of an algorithm is developed, that is used to generate these statements. As input, this algorithm needs an event log, and parameters need to be set. As output, sentences are provided. Parameters that can be set include, for example, what features to analyze (e.g. throughput time) and the linguistic labels for all features (e.g. when costs are considered as high).

This master thesis is not context specific, but focuses on business processes in general. Therefore, a general approach is chosen to determine what might be relevant to analyze for process data. Different protoforms and features are discussed that can be relevant in analyzing processes. However, for the scope of this thesis, not everything is investigated. The algorithm is designed in such a way that the framework can be extended easily.

The amount of data that is stored in an event log can be enormous and, therefore, the algorithm must be able to deal with this data efficiently. Even a minor choice can have a large influence on the running time. To achieve this, many implementation choices had to be made, such as: storing the data in an efficient way, an effective use of packages, and using pruning techniques if possible. In addition, the user can choose what, and what level of depth to analyze. These choices have a direct influence on the running time and, therefore, the user has to think about this thoroughly.

The evaluation shows that most people see the potential of linguistic summarization; how linguistic summarization can be used for the analysis of a business process and the benefits of using such a tool. All participants agree that the results that linguistic summarization provide, can be seen as an extension to the results of tools they currently use, and, therefore, they intend to use it in combination with the tools they currently use, e.g. use another tool for the visualization of the model. However, there are many improvements for the algorithm possible, such as: additional protoforms could be investigated, new features can be added, the user can be helped by setting all parameters, and the resulting set of summaries could be pruned even further.

Concluding, it seems that linguistic summarization can ease the analysis of (complex) processes, by providing new and interesting insights in natural language. However, the current technique can be extended and improved to provide more valuable insights, that are easier to understand.

10.2 Limitations

Although the algorithm that is created, can already be used in the analysis of process data, it is subjected to some limitations, discussed in this section.

Amount of results

The prototype that is created in this master thesis can result in many sentences to be validated, where not all sentences might be relevant for the user. As discussed in Section 8.2, the results are already pruned using several pruning techniques. However, some pruning steps are missing, that are based on the combinations of features or sequences of events. Some sentences that are currently shown, might give a wrong impression of the data and might, therefore, be seen as noise. For example, looking at the sentence *"In almost all cases, where Person A performed task Y, there was a large throughput time"* could give the impression that there is a *large throughput time* due to *Person A*. However, if the sentence *"In almost all cases, there was a large throughput time"* is also generated for the same process, the first sentence can be pruned, because the second sentence shows that there is a *large throughput time*, regardless of *Person A*.

Transactional life cycle

The transactional life-cycle model, as discussed in Section 5.2.1, is not investigated in this thesis. Event logs that contain the transactional life-cycle model can be investigated, but only the completions of the activities is taken into account.

Evaluation

Another limitation is that the evaluation is performed on only thirteen participants. Since this number is not sufficient enough to draw any conclusions, the evaluation only gives indications about the techniques' usefulness. It seems that linguistic summarization can be very useful in analyzing business processes, but this cannot be proven statistically.

Definition of linguistic labels

The last limitation discussed in this section is the creation of the linguistic labels, described in Section 6.2. For the scope of this thesis, it is chosen to let the user do this manually. Some statistics, e.g. the standard deviation and the average, are provided to the user, to give the user an indication about the distribution of the data. The linguistic labels can be based on these statistics. However, the creation of the linguistic labels can be hard and time consuming. This approach can be improved, for example, by visualizing the distribution of the data or by automatically proposing some linguistic labels, based on the data.

10.3 Future research

Next to the limitations discussed in previous section, there are some suggestions made for future research, which are discussed in this section.

Selected protoforms

Chapter 3 investigates the protoforms that might be relevant when analyzing process data. However, not all protoforms could be included for the scope of this thesis. This master thesis focuses on the protoforms that are most likely relevant for most processes, and, therefore, future research needs to determine in what manner the protoforms, not included in the scope, should be implemented.

Next to the additional protoforms that might be relevant, some protoforms can be combined, e.g. *"In most cases, where Person A was involved and a sequence like <ABC> was contained, there was a large throughput time"*, combines the case focused protoform for features related to the complete case and the case focused protoform for features related to the containment of sequences in the qualifier of the sentence.

Selected features

Next to the protoforms selected in the scope of this thesis, a choice had to be made regarding what features to analyze, which is discussed in Chapter 5. The list of features is based on event logs in general and is constructed to be as complete as possible. However, additional features might be relevant when processes are analyzed, such as the waiting time of a specific activity. In addition, as discussed in Section 5.4, the combinations of features are not implemented for unlimited attributes. Therefore, sentences like *"In most cases, where Person A was associated with high costs, there was a large throughput time"* cannot be created.

Selected parameters

In Section 6.4 and 7.1, some parameters are discussed, that can help in analyzing a business process. However, there might be additional parameters that are interesting to include, e.g. an upper limit for the degree of focus or a lower limit for the length of a sequence.

Selected validation measure

The truth value is used in this thesis as a validation measure for the linguistic summaries. However, there are more validation criteria, that might be relevant to include, e.g. criteria related to interestingness or usefulness for the user.

Resulting sentences

This research focuses on many different types of sentences that can be created. However, the structure of the sentences can be improved. For example, a combination between the resource '*Person A*' and activity '*Register appeal*' is shown as <|Resource = *Person A||Activity = Register appeal|*>, and can result in sentences like "*In most cases, there was* <|Resource = *Person A||Activity = Register appeal|*>". However, this might not be very clear at first sight, and the sentence "*In most cases, the resource Person A performed the activity Register appeal*" might be clearer.

Combining sentences

Sentences like *"In most cases, where Person A performs task X, the throughput time is large"* and *"In most cases, where Person B performs task X, the throughput time is large"* might be relevant when the throughput time is analyzed. However, a sentence like *"In most cases, where Person A or Person B performed task X, the throughput time is significantly larger than when other resources perform that task"* returns a clear overview of the situation within one sentence. This sentence does not necessary imply that there is a *large throughput time* when *Person A* or *Person B* performs *task X*, but at least there is a difference with other resources. The investigation and creation of such sentences is left for future research.

Visualization of results

The last topic discussed in this section focuses on the visualization of the sentences. Currently, the sentences are shown in natural language. However, there may be other possibilities that could return a clear, or clearer, overview, especially if many sentences are generated. A possible way to deal with the problem is by visualizing all features and their corresponding correlations, e.g. if the features *Person A performs task X* and *Person B performs task X* are positively correlated with the feature *large throughput time*, and the feature *Person C performs task X* is negatively correlated with the feature *large throughput time,* this can be visualized as shown in Figure 14, where green indicates a positive correlation and red means a negative correlation. The weight of the line can be used to represent the quantifier, e.g. a thick line represents *Almost all* and a thin line *Most*. However, there might be many features, and, therefore, many correlations, which can result in a so-called spaghetti model. To deal with this, there must be possibilities to filter the graph, based on, for example, the thickness of the arrows, the amount of connections of a feature, or a specific (set of) feature(s).

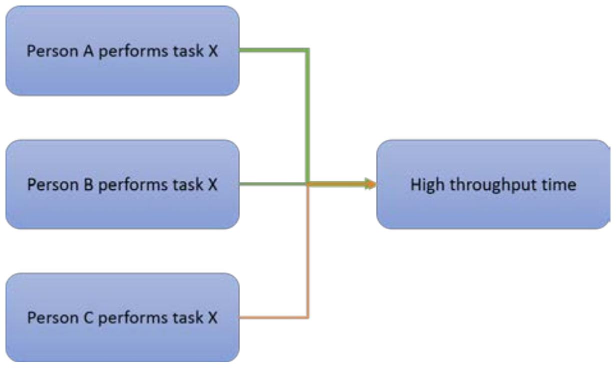


Figure 14: Visualization example

One could also choose to use the process model for the visualization, where the sentences are shown at the related activities, e.g. *Person A performs task X* is related to *task X.* By doing so, one can select a feature to be analyzed, which results in a process model where all activities are highlighted that contain sentences related to that feature, e.g. when the feature *large throughput time* is analyzed, and the sentence *"In most cases, where Person A performs task X, there is a large throughput time"* is contained in the linguistic summaries, *task X* is highlighted. By hovering over the activities, the corresponding sentences are shown. However, not all features can be linked to an activity, e.g. the feature *high costs* is related to the complete case. Therefore, this may be hard to visualize in this way.

10.4 Recommendations

The algorithm, developed in this master thesis, can already be used to get insights into a business process. Since this master thesis is not context specific, the technique can be used for all kind of processes and for different types of analysis, e.g. to get a high level overview of the business process or to analyze a specific (set of) feature(s). The algorithm is flexible to interact with, i.e. the user is able to select what has to be analyzed and what information can be discarded. In addition, the algorithm uses all information that is available in the event log and tries all combinations of features that may be worth analyzing, and is, therefore, highly suitable for event logs with many cases and/or many attributes. The results can already give surprising insights into the process, e.g. *"In almost all cases, when task X is performed by Person A, the throughput time of a sequence like <ABC> is large"*. It is hard, if not undoable, to get these kind of insights manually.

Results of the evaluation show that the technique has already a high potential to be useful for audit purposes. The technique is valuable in the automation of finding patterns in the data, that can be used to detect, for example, root causes of duplicate invoices. Based on the results, the client is possibly able to improve the process and prevent various duplicate purchase invoices in the future, which can reduce the work that the auditor has to do the year after. This can reduce the total costs of the audit process. In addition, the auditor could use the results of the research to get an indication where to focus on in finding such invoices. Therefore, KPMG is recommended to continue developing and using the proposed technique.

Bibliography

[1] W. M. P. van der Aalst, Process mining: discovery, conformance and enhancement of business processes, Heidelberg: Springer, 2011.

[2] W. M. P. van der Aalst and A. J. M. M. Weijters, "Process mining: a research agenda," *Computers in Industry,* pp. 231-244, 2004.

[3] F. Mannhardt, M. de Leoni, H. A. Reijers and W. M. P. van der Aalst, "Balanced multi-perspective checking of process conformance," *Computing,* vol. 4, no. 98, pp. 407-437, 2015.

[4] F. Mannhardt, M. de Leoni, H. A. Reijers and W. M. P. van der Aalst, "Decision Mining Revisited - Discovering Overlapping Rules," *BPM Center Report,* vol. 16, no. 1, pp. 1-17, 2016.

[5] M. de Leoni and W. M. P. van der Aalst, "Data-Aware Process Mining: Discovering Decisions in Processes Using Alignments," *Proceedings of the 28th Annual ACM Symposium on Applied Computing - SAC 13,* 2013.

[6] F. Mannhardt, M. de Leoni and H. A. Reijers, "Extending Process Logs with Events from Supplementary Sources," *Business Process Management Workshops Lecture Notes in Business Information Processing,* pp. 235-247, 2015.

[7] R. Conforti, M. de Leoni, M. La Rosa, W. M. P. van der Aalst, t. Hofstede and A. H. M, "A recommendation system for predicting risks across multiple business process instances," *Decision Support Systems,* vol. 69, pp. 1-19, 2015.

[8] J. Kacprzyk, R. R. Yager and S. Zadrożny, "Fuzzy Linguistic Summaries of Databases for an Efficient Business Data Analysis and Decision Support," *Knowledge Discovery for Business Information Systems The International Series in Engineering and Computer Science,* pp. 129-152, 2001.

[9] J. Kacprzyk and S. Zadrożny, "Linguistic database summaries and their protoforms: towards natural language based knowledge discovery tools," *Information Sciences,* vol. 173, no. 4, pp. 281-304, 2005.

[10] M. Ros, M. Pegalajar, M. Delgado, M. A. Vila, D. T. Anderson, J. M. Keller and M. Popescu, "Linguistic summarization of long-term trends for understanding change in human behavior," *2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011),* pp. 2080-2087, 2011.

[11] A. M. Wilbik, J. M. Keller and J. C. Bezdek, "Linguistic Prototypes for Data From Eldercare Residents," *IEEE Transactions on Fuzzy Systems,* vol. 22, no. 1, pp. 110 - 123, 2014.

[12] A. M. Wilbik, J. M. Keller and G. L. Alexander, "Linguistic summarization of sensor data for eldercare," *2011 IEEE International Conference on Systems, Man, and Cybernetics,* pp. 2595-2599, 2011.

[13] P. S. Szczepaniak and J. Ochelska, "Linguistic Summaries of Standardized Documents," *Studies in Computational Intelligence Advances in Web Intelligence and Data Mining,* pp. 221-232, 2006.

[14] J. Kacprzyk and A. M. Wilbik, "Towards an efficient generation of linguistic summaries of time series using a degree of focus," *NAFIPS 2009 - 2009 Annual Meeting of the North American Fuzzy Information Processing Society,* 2009.

[15] J. Kacprzyk, A. M. Wilbik and S. Zadrożny, "Linguistic summarization of time series using a fuzzy quantifier driven aggregation," *Fuzzy Sets and Systems,* vol. 159, no. 12, pp. 1485 - 1499, 2008.

[16] R. M. Castillo-Ortega, N. Marin and D. Sánchez, "A Fuzzy Approach to the Linguistic Summarization of Time Series," *Journal of Multiple-Valued Logic & Soft Computing,* vol. 17, pp. 157-182, 2011.

[17] R. M. Castillo-Ortega, N. Mann and D. Sánchez, "Linguistic local change comparison of time series," *2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011),* pp. 2909-2915, 2011.

[18] D. T. Anderson, R. H. Luke, J. M. Keller, M. Skubic, M. J. Rantz and M. A. Aud, "Modeling Human Activity From Voxel Person Using Fuzzy Logic," *IEEE Transactions on Fuzzy Systems,* vol. 17, no. 1, pp. 39-49, 2009.

[19] D. T. Anderson, R. H. Luke, J. M. Keller, M. Skubic, M. Rantz and M. A. Aud, "Linguistic summarization of video for fall detection using voxel person and fuzzy logic," *Computer Vision and Image Understanding,* vol. 113, no. 1, pp. 80-89, 2009.

[20] D. T. Anderson, R. H. Luke and J. M. Keller, "Segmentation and linguistic summarization of voxel environments using stereo vision and genetic algorithms," *International Conference on Fuzzy Systems,* 2010.

[21] S. Zadrożny and J. Kacprzyk, "Summarizing the Contents of Web Server Logs: A Fuzzy Linguistic Approach," *2007 IEEE International Fuzzy Systems Conference,* 2007.

[22] A. M. Wilbik and U. Kaymak, "Linguistic Summarization of Processes – a research agenda," *Proceedings of the 2015 Conference of the International Fuzzy Systems Association and the European Society for Fuzzy Logic and Technology,* 2015.

[23] A. M. Wilbik and R. M. Dijkman, "Linguistic summaries of process data," *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE),* 2015.

[24] A. M. Wilbik and R. M. Dijkman, "On the generation of useful linguistic summaries of sequences," *2016 IEEE World Congress on Computational Intelligence (WCCI),* 2016.

[25] Python Software Foundation, "https://www.python.org/," Python Software Foundation, 2017. [Online]. Available: https://www.python.org/. [Accessed 10 02 2017].

[26] A. R. Hevner, S. T. March, P. J and R. S, "Design Science in Information Systems Research," *MIS Quarterly,* vol. 24, no. 1, pp. 75-105, 2004.

[27] G. P. Hodgkinson and D. M. Rousseau, "Bridging the Rigour-Relevance Gap in Management Research: Its Already Happening!," *Journal of Management Studies,* vol. 46, no. 3, pp. 534-546, 2009.

[28] R. M. Dijkman, M. Dumas and C. Ouyang, "Semantics and analysis of business process models in BPMN," *Information and Software Technology,* vol. 50, no. 12, pp. 1281-1294, 2008.

[29] R. R. Yager, "A New Approach to the Summarization of Data," *Information Sciences,* vol. 28, no. 2, pp. 69-86, 1982.

[30] C. Carlsson and R. Fullér, "Possibility for decision: a possibilistic approach to real life decisions," in *Concepts and Issues*, Berlin, Springer, 2013, p. 20.

[31] L. A. Zadeh, "Fuzzy Sets," *Information and Control,* vol. 8, no. 3, pp. 338-253, 1965.

[32] L. A. Zadeh, "A prototype-centered approach to adding deduction capability to search engines-the concept of protoform," *Proceedings First International IEEE Symposium Intelligent Systems,* 2002.

[33] M. Delgado, M. D. Ruiz, D. Sánchez and M. A. Vila, "Fuzzy quantification: a state of the art," *Fuzzy Sets and Systems,* vol. 242, pp. 1-30, 2014.

[34] A. Jain and J. M. Keller, "On the computation of semantically ordered truth values of linguistic protoform summaries," *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE),* 2015.

[35] A. M. Wilbik, U. Kaymak, J. M. Keller and M. Popescu, "Evaluation of the Truth Value of Linguistic Summaries – Case with Non-monotonic Quantifiers," *Advances in Intelligent Systems and Computing Intelligent Systems'2014,* pp. 69-79, 2015.

[36] F. E. Boran, D. Akay and R. R. Yager, "An overview of methods for linguistic summarization with fuzzy sets," *Expert Systems with Applications,* vol. 61, pp. 356-377, 2016.

[37] L. A. Zadeh, "Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic," *Fuzzy Sets and Systems,* vol. 90, no. 2, pp. 111-127, 1997.

[38] A. M. Wilbik and J. Kacprzyk, "On the evaluation of the linguistic summarization of temporally focused time series using a measure of informativeness," *Proceedings of the International Multiconference on Computer Science and Information Technology,* pp. 155-162, 2010.

[39] D. Dubois and H. Prade, "Gradual inference rules in approximate reasoning," *Information Sciences,* vol. 61, no. 1-2, pp. 103-122, 1992.

[40] D. Dubois, H. Prade and E. Rannou, "User-driven summarization of data based on gradual rules," *Proceedings of 6th International Fuzzy Systems Conference,* vol. 2, pp. 839-844, 1997.

[41] G. Raschia and N. Mouaddib, "SAINTETIQ: a fuzzy set-based approach to database summarization," *Fuzzy Sets and Systems,* vol. 129, no. 2, pp. 137-162, 2002.

[42] M. R. Rajati and J. M. Mendel, "Advanced computing with words using syllogistic reasoning and arithmetic operations on linguistic belief structures," *2013 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE),* 2013.

[43] G. Moyse, M.-J. Lesot and B. Bouchon-Meunier, "Linguistic summaries for periodicity detection based on mathematical morphology," *IEEE Symposium Series on Computational Intelligence,* pp. 106-113, 2013.

[44] G. Moyse, M.-J. Lesot and B. Bouchon-Meunier, "Mathematical Morphology Tools to Evaluate Periodic Linguistic Summaries," *Flexible Query Answering Systems Lecture Notes in Computer Science,* pp. 257-268, 2013.

[45] H. M. Cooper, L. V. Hedges and J. C. Valentine, The handbook of research synthesis and meta-analysis, New York: Russell Sage Foundation, 2009.

[46] B. F. v. Dongen, "Process mining and verification," PhD thesis, Technische Universiteit Eindhoven, 2007.

[47] A. L. Medaglia, S.-C. Fang, H. L. W. Nuttle and J. R. Wilson, "An efficient and flexible mechanism for constructing membership functions," *European Journal of Operational Research,* vol. 139, no. 1, pp. 84-95, 2002.

[48] MathWorks, "nl.mathworks.com," MathWorks, 2017. [Online]. Available: https://nl.mathworks.com/help/fuzzy/trapmf.html. [Accessed 15 05 2017].

[49] R. M. Dijkman, M. Dumas, B. F. v. Dongen, R. Käärik and J. Mendling, "Similarity of business process models: Metrics and evaluation," *Information Systems,* vol. 36, no. 2, pp. 498-516, 2011.

[50] The Scipy community, "https://docs.scipy.org," 11 05 2014. [Online]. Available: https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.cluster.hierarchy.linkage.html. [Accessed 02 03 2017].

[51] The Scipy community., "The Scipy community," https://docs.scipy.org, 18 01 2015. [Online]. Available: https://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.cluster.hierarchy.fcluster.html. [Accessed 02 03 2017].

[52] M. Popescu, J. C. Bezdek, T. C. Havens and J. M. Keller, "A Cluster Validity Framework Based on Induced Partition Dissimilarity," *IEEE Transactions on Cybernetics,* vol. 43, no. 1, pp. 308-320, 2013.

[53] J. C. Bezdek and R. J. Hathaway, "VAT: a tool for visual assessment of (cluster) tendency," *Proceedings of the 2002 International Joint Conference on Neural Networks,* pp. 2225-22230, 2002.

[54] T. C. Havens and J. C. Bezdek, "An Efficient Formulation of the Improved Visual Assessment of Cluster Tendency (iVAT) Algorithm," *IEEE Transactions on Knowledge and Data Engineering,* vol. 24, no. 5, pp. 813-822, 2012.

[55] D. L. Davies and D. W. Bouldin, "A Cluster Separation Measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vols. PAMI-1, no. 2, pp. 224-227, 1979.

[56] X. L. Xie and G. Beni, "A validity measure for fuzzy clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 13, no. 8, pp. 841-847, 1991.

[57] A. M. Wilbik, U. Kaymak and R. M. Dijkman, "Towards improved generation of linguistic summaries," *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) (in press),* 2017.

[58] Fluxicon Process Laboratories, "https://fluxicon.com," Fluxicon Process Laboratories, 2012. [Online]. Available: https://fluxicon.com/disco/. [Accessed 10 05 2017].

[59] D. A. Adams, R. R. Nelson and P. A. Todd, "Perceived Usefulness, Ease of Use, and Usage of Information Technology: A Replication," *MIS Quarterly,* vol. 16, no. 2, pp. 227-247, 1992.

[60] F. D. Davis, "Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology," *MIS Quarterly,* vol. 13, no. 3, pp. 319-340, 1989.

[61] V. Venkatesh and F. D. Davis, "A Theoretical Extension of the Technology Acceptance Model: Four Longitudinal Field Studies," *Management Science,* vol. 46, no. 2, pp. 186-204, 2000.

[62] R. M. Dijkman and A. M. Wilbik, "Linguistic Summarization of Event Logs - A Practical Approach," *Preprint submitted to Information Systems,* pp. 1-42, 2017.

Appendix A: User manual

As part of this thesis, a tool is implemented, in Python, that can be used to create linguistic summaries about an event log. To ease the interaction with the tool, this user manual is created, that guides the user through the process. The user manual is structured similar to the process flow, and includes the following steps:

1) Select event log;
2) Select delimiter;
3) Select or create dictionary;
4) Specify column definitions;
5) Select features to analyze;
6) Specify linguistic labels;
7) Select parameters;
8) Select clustering;
9) Specify linguistic labels for times related to sequences;
10) Scope the analysis;
11) Get results.

These steps are elaborated next.

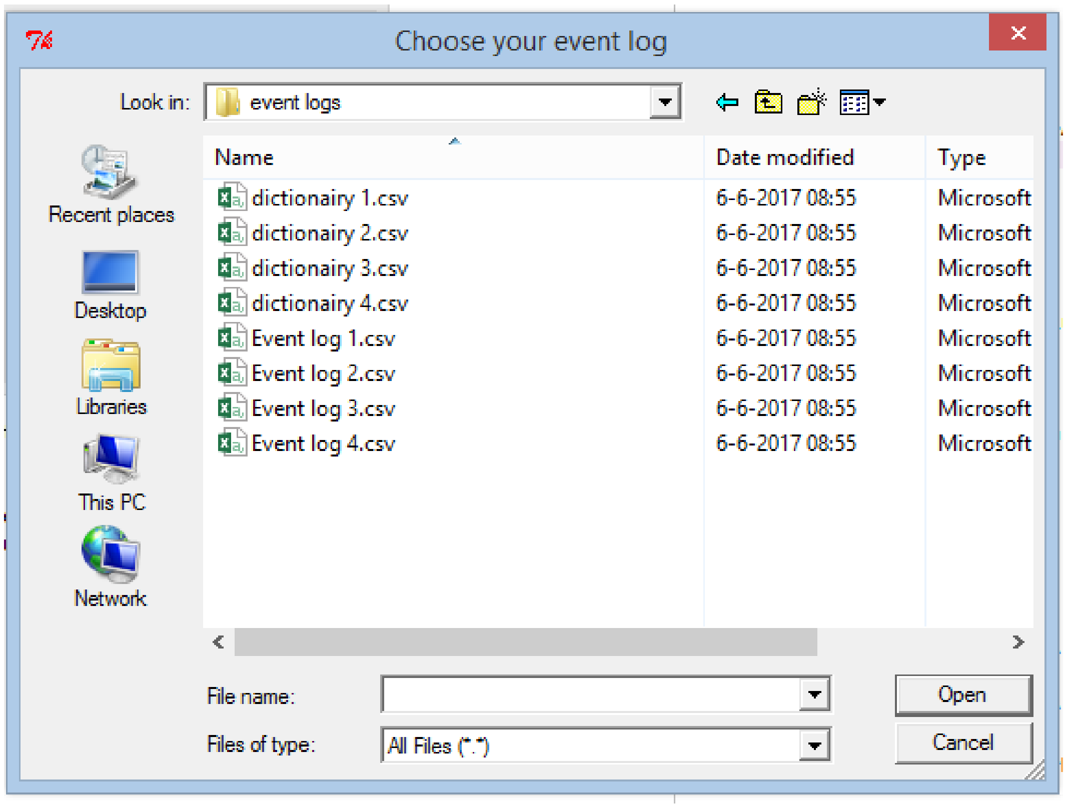
1) Select event log

When the program is started, the user is asked to select the event log to be analyzed, visualized in Screen 1. The event log has to be in csv format. When the user selects a file that is not in csv format, the user is asked whether another file has to be selected, or whether the program has to be terminated, shown in Screen 2.

<u>Assumption 1:</u> The event log is first ordered on case identifier and then on timestamp. This is done to make sure that the ordering of the activities is done as the user wants.

<u>Assumption 2:</u> The case attributes (attributes that remain the same for the complete case), as described in Section 5.3, are set at the first activity of every case.

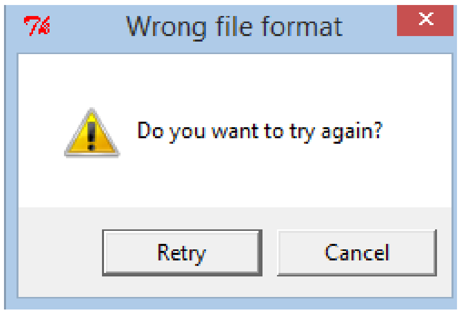
<u>Assumption 3:</u> The case identifier and timestamps, if applicable, are always filled, i.e. these columns do not contain any missing data.



Name	Date modified	Type
dictionairy 1.csv	6-6-2017 08:55	Microsoft
dictionairy 2.csv	6-6-2017 08:55	Microsoft
dictionairy 3.csv	6-6-2017 08:55	Microsoft
dictionairy 4.csv	6-6-2017 08:55	Microsoft
Event log 1.csv	6-6-2017 08:55	Microsoft
Event log 2.csv	6-6-2017 08:55	Microsoft
Event log 3.csv	6-6-2017 08:55	Microsoft
Event log 4.csv	6-6-2017 08:55	Microsoft

File name:
Files of type: All Files (*.*)

Screen 1: Select event log



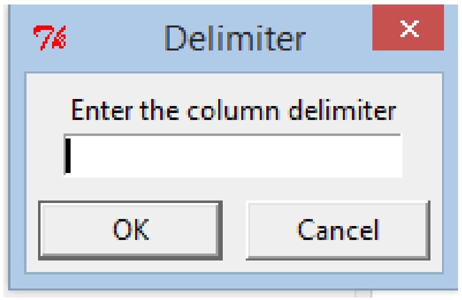
Do you want to try again?

Screen 2: error when no csv format is selected

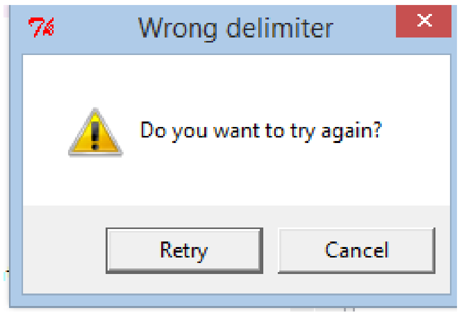
2) Select delimiter

Next, the delimiter has to be specified, which is used to separate the columns, see Screen 3. The delimiter is a single character. A comma cannot be used as a delimiter, since the delimiter is used in later steps, to store the results. When a wrong delimiter is chosen, the user is asked whether the user wants to try another delimiter, or whether the program can be terminated, see Screen 4. A wrong delimiter is identified when:

- It contains more than 1 character;
- The delimiter is empty;
- The number of columns of the first row, that is used to specify the names of the columns, is unequal to the number of columns of the second row, that contains the first row of data.



Screen 3: Specify delimiter



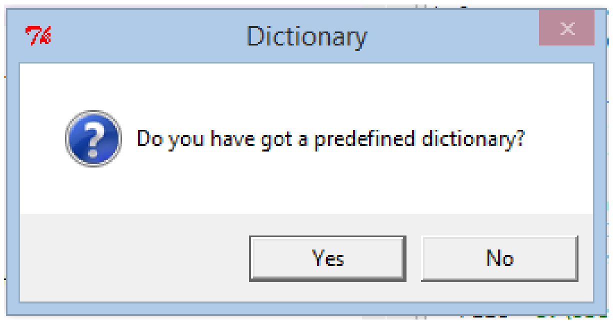
Do you want to try again?

Screen 4: Error when wrong delimiter is chosen

3) Select or create dictionary

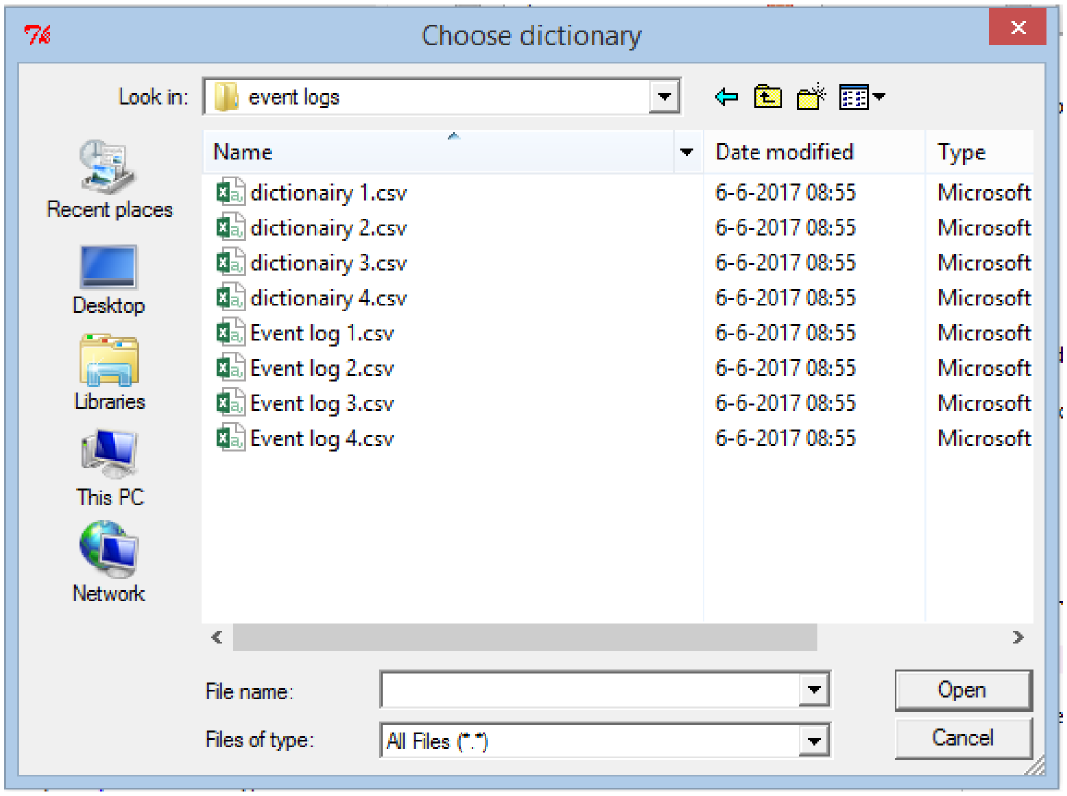
The first time the process is executed, the user has to enter many details about the event log, e.g. what has to be analyzed. Since the user probably wants to run the program more than once, the option to store all details in a so-called dictionary is given. After the event log is selected and the delimiter is set, the user is asked whether this dictionary is already created, see Screen 5. If 'Yes' is selected, i.e. there exists a dictionary, the user is asked to select the dictionary, see Screen 6. If not, the user is asked where the 'new' dictionary has to be stored, see Screen 7.

Note: When a dictionary is selected, the user can follow the same steps, where the settings are already filled in. Changes in the configuration will result in changes in the dictionary. A backup of the dictionary can be created to make sure a correct version is kept.



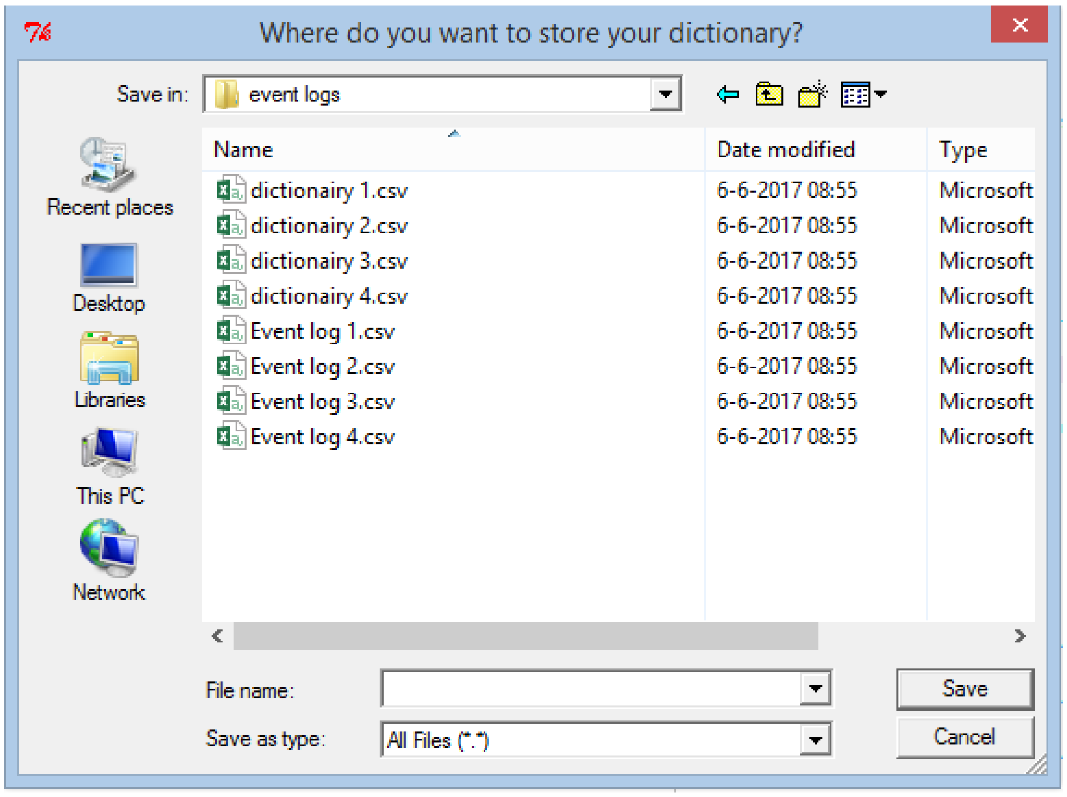
Do you have got a predefined dictionary?

Screen 5: Question whether dictionary is already created



Name	Date modified	Type
dictionairy 1.csv	6-6-2017 08:55	Microsoft
dictionairy 2.csv	6-6-2017 08:55	Microsoft
dictionairy 3.csv	6-6-2017 08:55	Microsoft
dictionairy 4.csv	6-6-2017 08:55	Microsoft
Event log 1.csv	6-6-2017 08:55	Microsoft
Event log 2.csv	6-6-2017 08:55	Microsoft
Event log 3.csv	6-6-2017 08:55	Microsoft
Event log 4.csv	6-6-2017 08:55	Microsoft

Screen 6: Choose dictionary



Name	Date modified	Type
dictionairy 1.csv	6-6-2017 08:55	Microsoft
dictionairy 2.csv	6-6-2017 08:55	Microsoft
dictionairy 3.csv	6-6-2017 08:55	Microsoft
dictionairy 4.csv	6-6-2017 08:55	Microsoft
Event log 1.csv	6-6-2017 08:55	Microsoft
Event log 2.csv	6-6-2017 08:55	Microsoft
Event log 3.csv	6-6-2017 08:55	Microsoft
Event log 4.csv	6-6-2017 08:55	Microsoft

Screen 7: Select location to store dictionary

4) Specify column definitions

To make sure the analysis is based on the correct data, the user is asked to specify the column definitions. When this step has not been taken previously, i.e. when no dictionary is selected, all columns must be specified from scratch, see Screen 8. The algorithm identifies the different column names and shows the amount of unique values for that column, e.g. there are 1268 different values for the column 'Case ID', which indicates that the log consists of 1268 cases.

In the second column, the column types can be filled. For the column type, the following options are available:

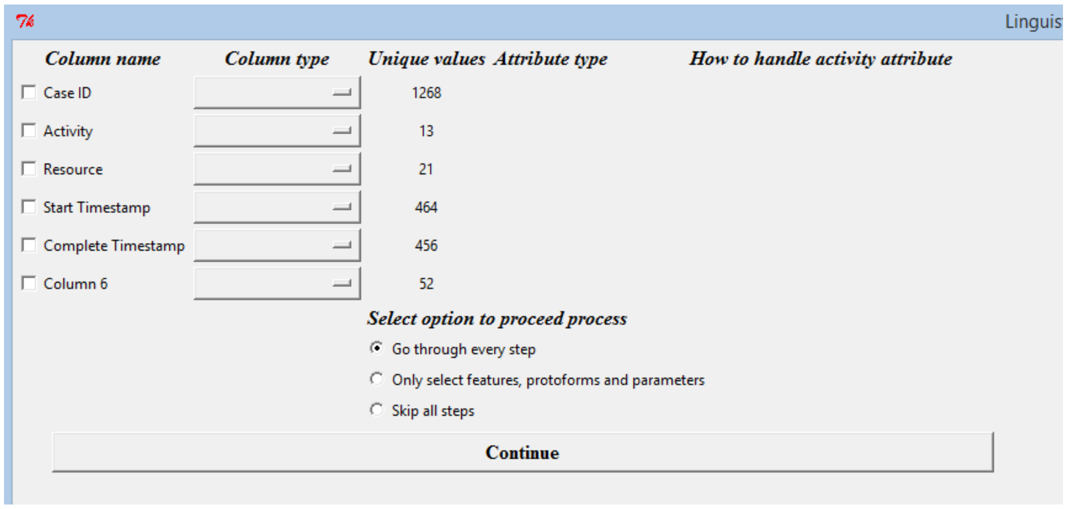
- Activity;
- Activity Attribute;
- Case Attribute;
- Case identifier;
- End timestamp;
- Start timestamp;
- State in lifecycle.

These options are further explained in Chapter 5.

When an attribute is selected (either an activity attribute or a case attribute), the user must specify, in the fourth column, whether it is a limited or unlimited attribute. In addition, when an activity attribute is selected, the user must specify how to handle the attribute:

- Separately (different membership function per activity): e.g. a fine is created with the activity *Create Fine*. When the user does not pay the fine within one month, a penalty is added with the activity *Add penalty*. It can be the case that a fine of €100 is considered as low, but a penalty of €100 is considered as high. When this is the case, different membership functions need to be created for every activity that is related to this amount. By treating the activities related to the attribute separately, sentences can be created about when the attribute belongs to a certain set for a certain activity, e.g. *the Amount is high for Create fine*;
- Separately (Same membership function per activity): It is not necessary the case that different membership functions are needed for every activity. Considering the example shown above, it could be the case that both a fine and penalty are high for the same amount;
- Together (Same membership function): If the user does not want to distinguish between the activities that are related to that attribute, sentences can be about the attribute in general, e.g. *the Amount was high*, instead of *the Amount is high for Create fine*.

For more details about these types of attributes, see Section 5.3.



Linguis

Column name	Column type	Unique values	Attribute type	How to handle activity attribute
☐ Case ID		1268		
☐ Activity		13		
☐ Resource		21		
☐ Start Timestamp		464		
☐ Complete Timestamp		456		
☐ Column 6		52		

Select option to proceed process

[✓ Go through every step]

Only select features, protoforms and parameters

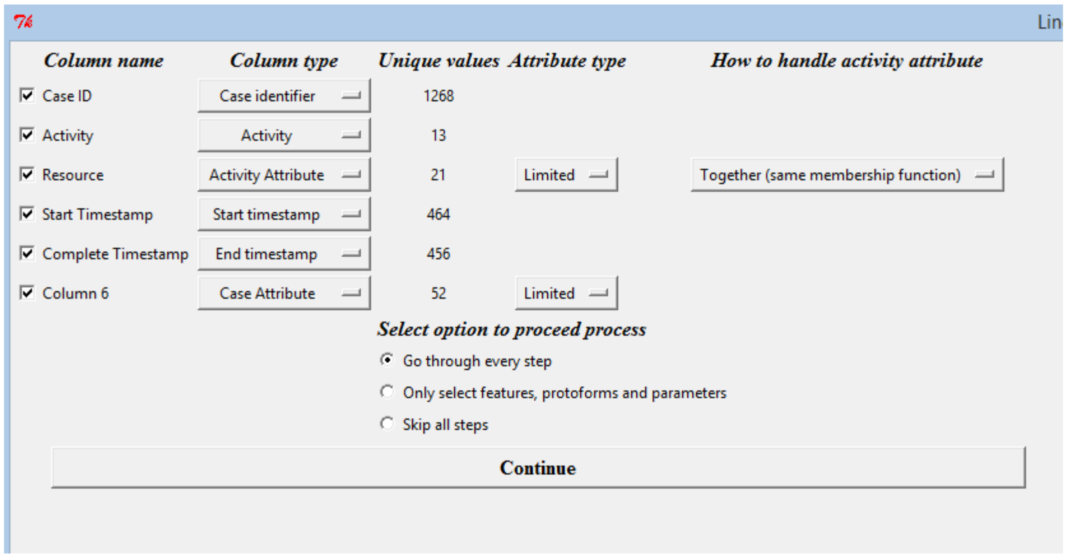
Skip all steps

Screen 8: Specify column definitions

Screen 9 shows an example of a screen where the column definitions are filled in. Finally, the user can choose how to proceed:

- Go through every step: This option can be selected when the process is done for the first time or when (new) membership functions need to be created and/or changed;
- Only select features, protoforms and parameters: By choosing this option, the definitions of the membership functions is skipped. However the user is still able to select what to analyze;
- Skip all steps: Using this step, no settings can be changed, and the user is directed to Step 10).

Note that the second and third option can only be selected when the membership functions are already defined before.



Lin

Column name	Column type	Unique values	Attribute type	How to handle activity attribute
✓ Case ID	Case identifier	1268		
✓ Activity	Activity	13		
✓ Resource	Activity Attribute	21	Limited	Together (same membership function)
✓ Start Timestamp	Start timestamp	464		
✓ Complete Timestamp	End timestamp	456		
✓ Column 6	Case Attribute	52	Limited	

Select option to proceed process

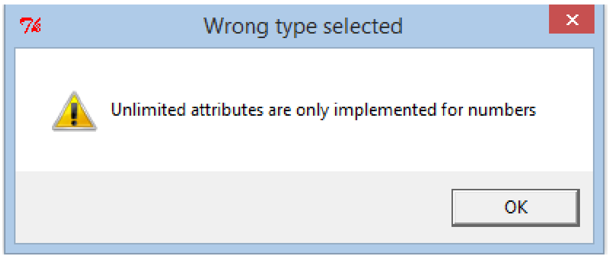
[✓ Go through every step]

Only select features, protoforms and parameters

Skip all steps

Screen 9: Specify column definitions filled

Note: When an attribute is selected to be 'unlimited', it is checked whether the column can be converted to numbers. When this is not the case, no membership function can be created, and the error shown in Screen 10 is shown. The user will be redirected to Screen 9.



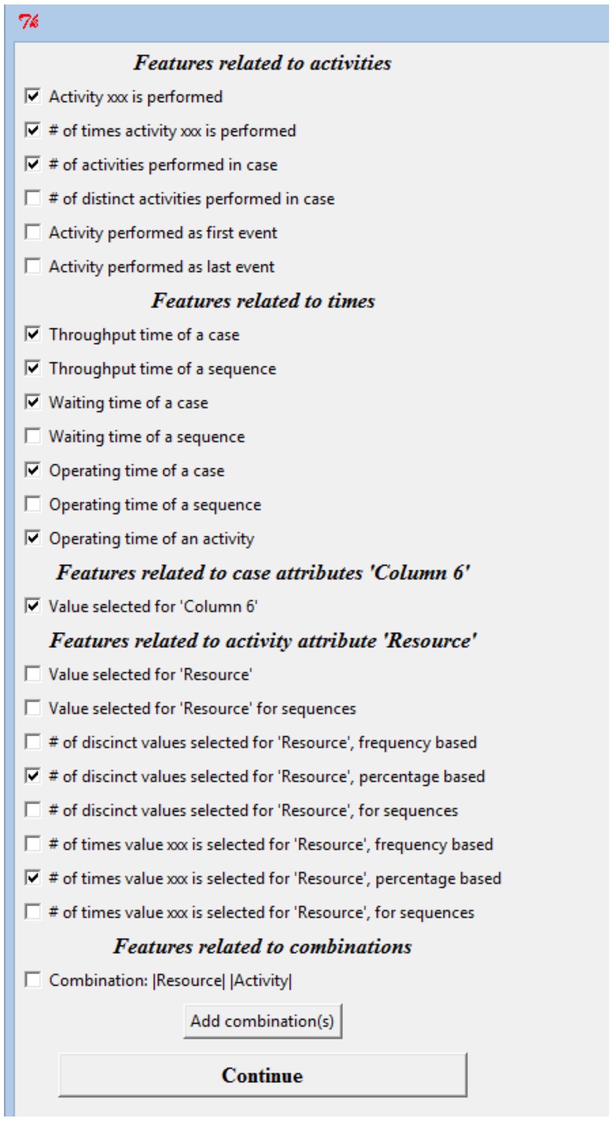
Unlimited attributes are only implemented for numbers

Screen 10: Error message unlimited attributes

5) Select features to analyze

After the definitions of all columns are specified, the features can be selected that the user wants to analyze, which is shown in Screen 11. The algorithm proposes a set of features that can be analyzed, based on the column definitions specified in the previous step, e.g. if there is no end timestamp of an activity, the operating time of an activity cannot be analyzed. More information on this topic can be found in Chapter 5. The screen to select the combinations of features, as discussed in Section 5.4 is shown in Screen 12. One can create up to five combinations of at most five features to use in the analysis.

Note: There are several features related to the timestamps. The unit of these features is chosen to be in days, but can easily be converted to other formats. The timestamp can be logged in many different ways, e.g. 2016-07-01 or 01-07-2016 00:00:00. To be able to automatically identify the timestamp format, the parse method is used (with default parameters) from the parser module from the dateutil library, see http://dateutil.readthedocs.io/en/stable/parser.html for more details.


Features related to activities

- [✓] Activity xxx is performed
- [✓] # of times activity xxx is performed
- [✓] # of activities performed in case
- [ ] # of distinct activities performed in case
- [ ] Activity performed as first event
- [ ] Activity performed as last event

Features related to times

- [✓] Throughput time of a case
- [✓] Throughput time of a sequence
- [✓] Waiting time of a case
- [ ] Waiting time of a sequence
- [✓] Operating time of a case
- [ ] Operating time of a sequence
- [✓] Operating time of an activity

Features related to case attributes 'Column 6'

- [✓] Value selected for 'Column 6'

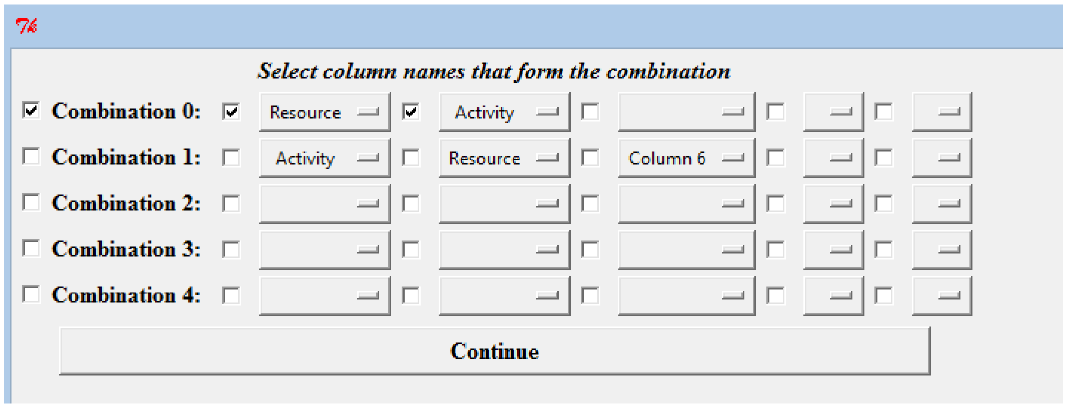
Features related to activity attribute 'Resource'

- [ ] Value selected for 'Resource'
- [ ] Value selected for 'Resource' for sequences
- [ ] # of discinct values selected for 'Resource', frequency based
- [✓] # of discinct values selected for 'Resource', percentage based
- [ ] # of discinct values selected for 'Resource', for sequences
- [ ] # of times value xxx is selected for 'Resource', frequency based
- [✓] # of times value xxx is selected for 'Resource', percentage based
- [ ] # of times value xxx is selected for 'Resource', for sequences

Features related to combinations

- [ ] Combination: |Resource| |Activity|

Screen 11: Feature selection

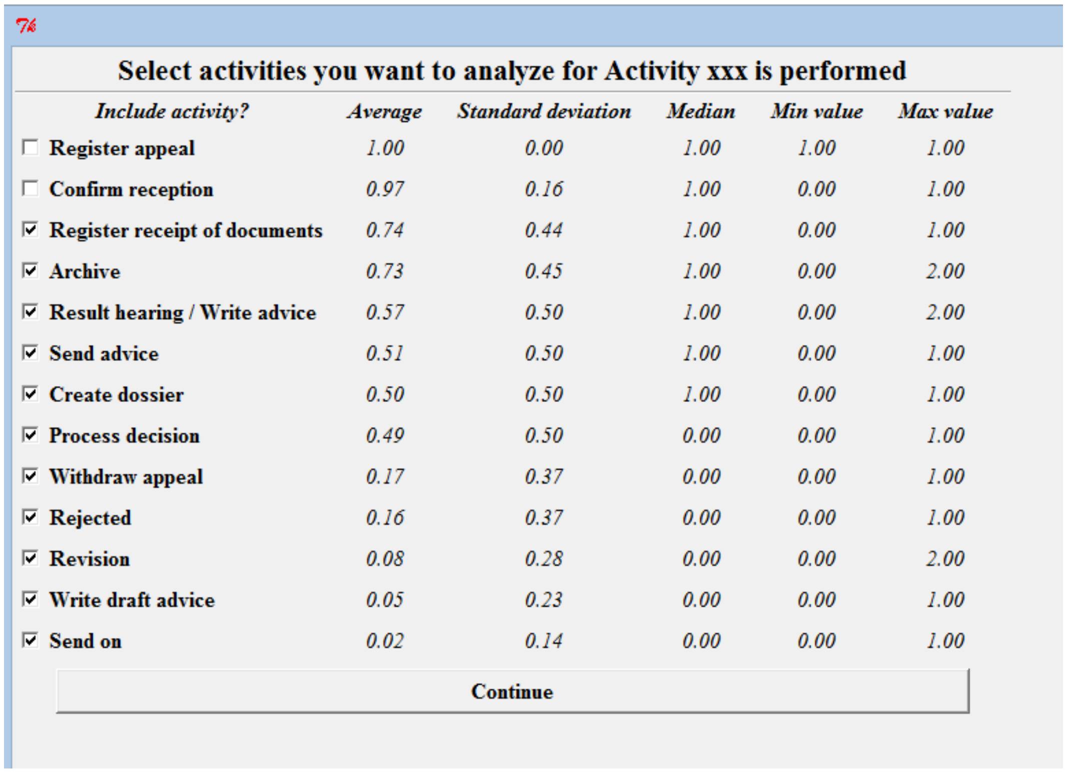

Select column names that form the combination

[✓] **Combination 0:**	[✓]	Resource	[✓]	Activity	[ ]		[ ]		[ ]	
[ ] **Combination 1:**	[ ]	Activity	[ ]	Resource	[ ]	Column 6	[ ]		[ ]	
[ ] **Combination 2:**	[ ]		[ ]		[ ]		[ ]		[ ]	
[ ] **Combination 3:**	[ ]		[ ]		[ ]		[ ]		[ ]	
[ ] **Combination 4:**	[ ]		[ ]		[ ]		[ ]		[ ]	

Screen 12: Specify combinations

For some features, user input is needed, that can be used as a scope for the analysis:

- *Activity xxx is performed*: Screen 13 shows how the user can scope the analysis when this feature is selected. Some statistics are given about the occurrence of all activities, which can be used to scope the analysis, e.g. the activity '*Register*' appeal occurred, on average, once per case, with a standard deviation of zero, it occurred at least once and at most once per case, and (logically), the median is also one. It can be the case that the user does not want to analyze all features, e.g. because it occurred in every case exactly once. Filtering this feature does not have any influence on other features selected;
- *# of times activity xxx is performed*: Also for this feature, some statistics are shown that can be used to build the scope, e.g. when the activity only occurred once, this feature probably does not make any sense, since it is contained in the feature *Activity xxx is performed*;
- *Operating time of an activity*: The last feature that needs user input is the feature *Operating time of an activity*. Some statistics are shown about the operating time of every activity, that can be used to set the scope, e.g. only analyze the activities with the highest average operating time, or only analyze the activities where the operating time differs a lot, i.e. where the standard deviation is high.



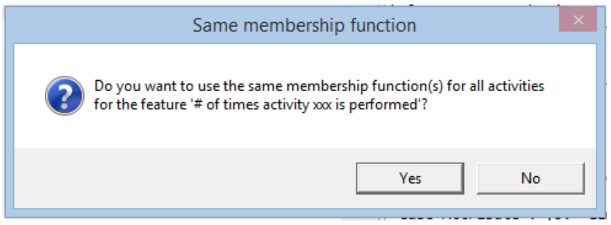
Select activities you want to analyze for Activity xxx is performed

Include activity?	*Average*	*Standard deviation*	*Median*	*Min value*	*Max value*
☐ Register appeal	1.00	0.00	1.00	1.00	1.00
☐ Confirm reception	0.97	0.16	1.00	0.00	1.00
☑ Register receipt of documents	0.74	0.44	1.00	0.00	1.00
☑ Archive	0.73	0.45	1.00	0.00	2.00
☑ Result hearing / Write advice	0.57	0.50	1.00	0.00	2.00
☑ Send advice	0.51	0.50	1.00	0.00	1.00
☑ Create dossier	0.50	0.50	1.00	0.00	1.00
☑ Process decision	0.49	0.50	0.00	0.00	1.00
☑ Withdraw appeal	0.17	0.37	0.00	0.00	1.00
☑ Rejected	0.16	0.37	0.00	0.00	1.00
☑ Revision	0.08	0.28	0.00	0.00	2.00
☑ Write draft advice	0.05	0.23	0.00	0.00	1.00
☑ Send on	0.02	0.14	0.00	0.00	1.00

Screen 13: Select which activities to analyze for *Activity xxx is performed*

6) Specify linguistic labels

For some features, linguistic labels have to be defined, e.g. the definition of a *large throughput time*. More information on this topic can be found in 6.2. Some features can, for certain processes, be represented by the same membership functions, e.g. the different activities selected for the feature *# of times activity xxx is performed*. When features are selected that could be represented by the same membership function, the user is asked whether this is needed, as shown in Screen 14.

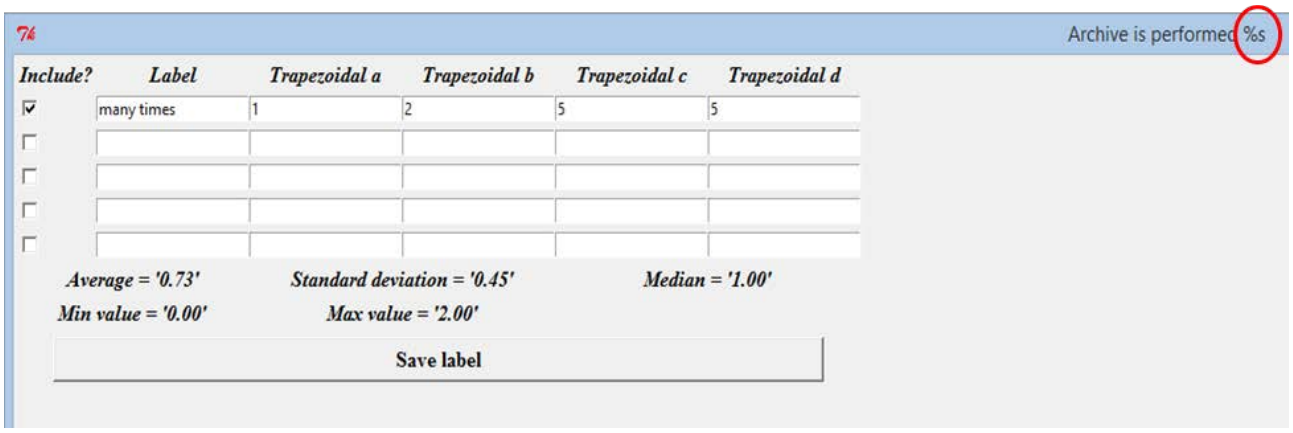


Same membership function

Do you want to use the same membership function(s) for all activities for the feature '# of times activity xxx is performed'?

Screen 14: Question whether same membership functions are needed for a feature

The definitions of the linguistic labels is done as shown in Screen 15 and Screen 16. For some features, a template is created and the user only has to fill in the gap, marked with %s. This is shown in Screen 15. For other features, the label has to be fully defined by the user, which is shown in Screen 16.

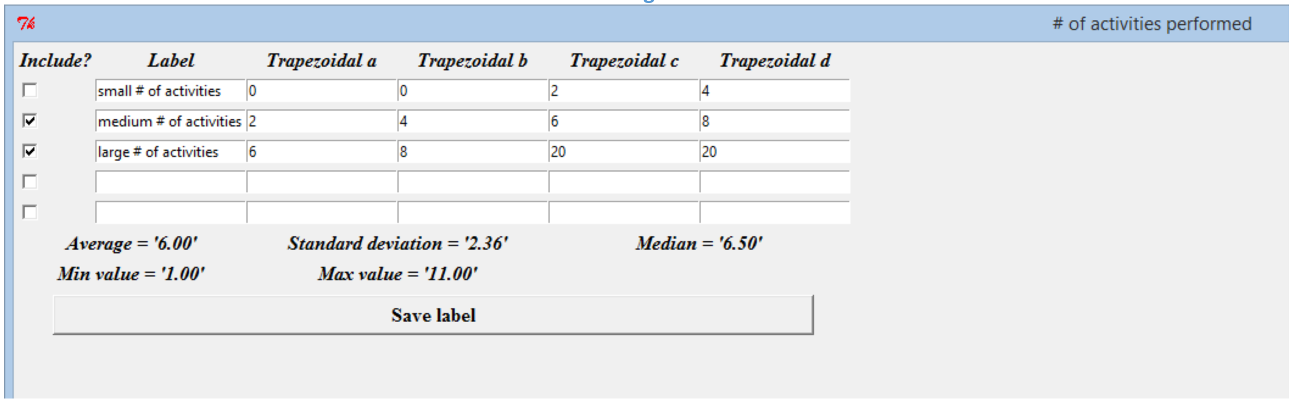


Archive is performed %s

Include?	Label	Trapezoidal a	Trapezoidal b	Trapezoidal c	Trapezoidal d
✓	many times	1	2	5	5

Average = '0.73' Standard deviation = '0.45' Median = '1.00'
Min value = '0.00' Max value = '2.00'

Screen 15: Create linguistic labels with %s



# of activities performed

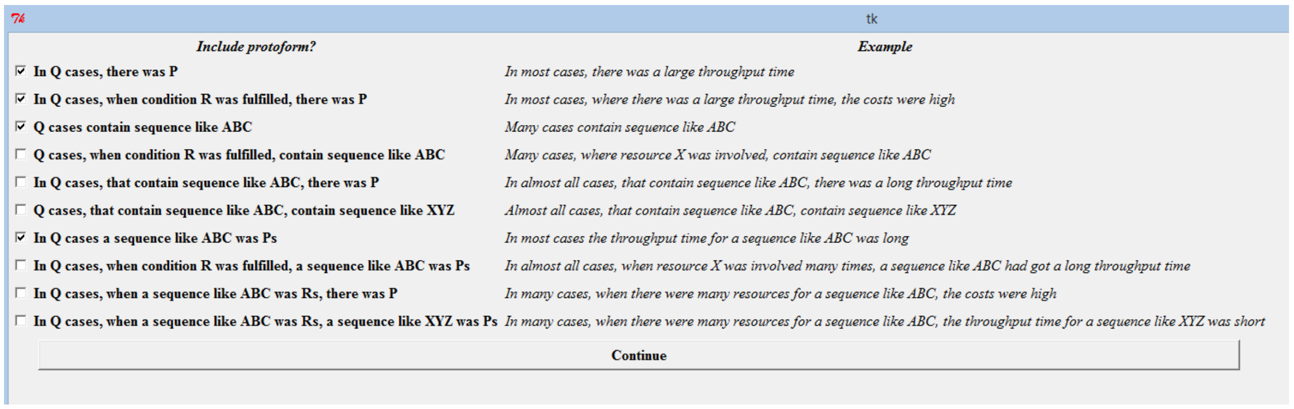
Include?	Label	Trapezoidal a	Trapezoidal b	Trapezoidal c	Trapezoidal d
	small # of activities	0	0	2	4
✓	medium # of activities	2	4	6	8
✓	large # of activities	6	8	20	20

Average = '6.00' Standard deviation = '2.36' Median = '6.50'
Min value = '1.00' Max value = '11.00'

Screen 16: Create linguistic labels without %s

7) Select parameters

After all linguistic labels are filled in, the parameters are set. First, the protoforms are selected that the user wants to analyze, shown in Screen 17. More information about all protoform can be found in Chapter 3.



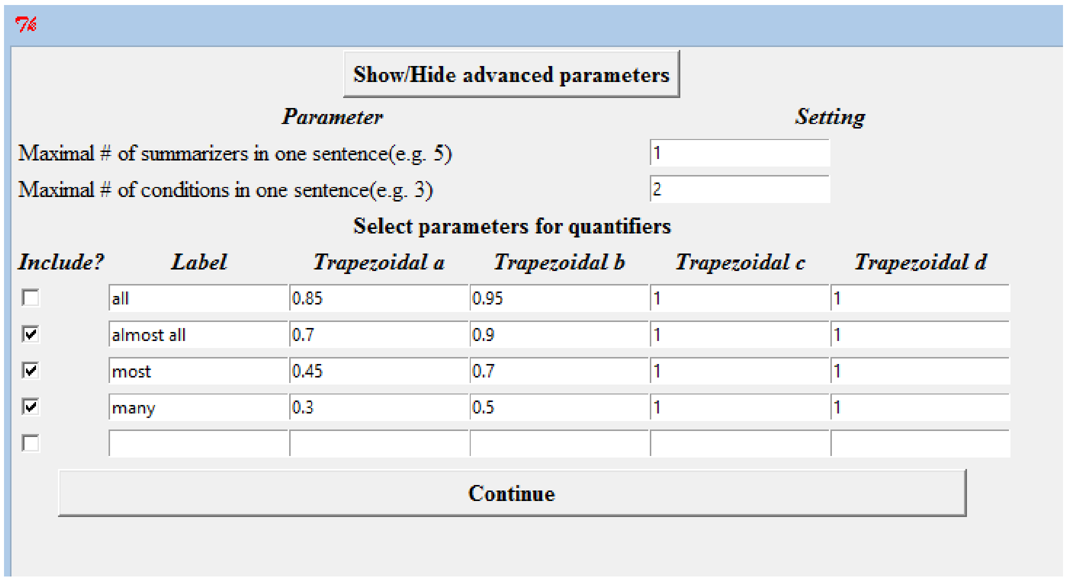
tk

Include protoform?	Example
✓ In Q cases, there was P	*In most cases, there was a large throughput time*
✓ In Q cases, when condition R was fulfilled, there was P	*In most cases, where there was a large throughput time, the costs were high*
✓ Q cases contain sequence like ABC	*Many cases contain sequence like ABC*
☐ Q cases, when condition R was fulfilled, contain sequence like ABC	*Many cases, where resource X was involved, contain sequence like ABC*
☐ In Q cases, that contain sequence like ABC, there was P	*In almost all cases, that contain sequence like ABC, there was a long throughput time*
☐ Q cases, that contain sequence like ABC, contain sequence like XYZ	*Almost all cases, that contain sequence like ABC, contain sequence like XYZ*
✓ In Q cases a sequence like ABC was Ps	*In most cases the throughput time for a sequence like ABC was long*
☐ In Q cases, when condition R was fulfilled, a sequence like ABC was Ps	*In almost all cases, when resource X was involved many times, a sequence like ABC had got a long throughput time*
☐ In Q cases, when a sequence like ABC was Rs, there was P	*In many cases, when there were many resources for a sequence like ABC, the costs were high*
☐ In Q cases, when a sequence like ABC was Rs, a sequence like XYZ was Ps	*In many cases, when there were many resources for a sequence like ABC, the throughput time for a sequence like XYZ was short*

Screen 17: Select protoforms to analyze

Next to the protoforms that are selected, different parameters can be set. This screen only shows the relevant parameters, e.g. when no qualifier is needed, the degree of focus does not have to be set. Screen 18 shows the basic parameters, and Screen 19 shows all, basic and advanced, parameters. One can choose to hide the advanced parameters, for simplicity. All parameters are explained in more detail in Section 6.4 for the generic parameters, and in Section 7.1 for the parameters related to sequences.

For every parameter, some default settings are chosen. However, since different aspects need to be analyzed for every event log, these parameters need to be changed possibly.

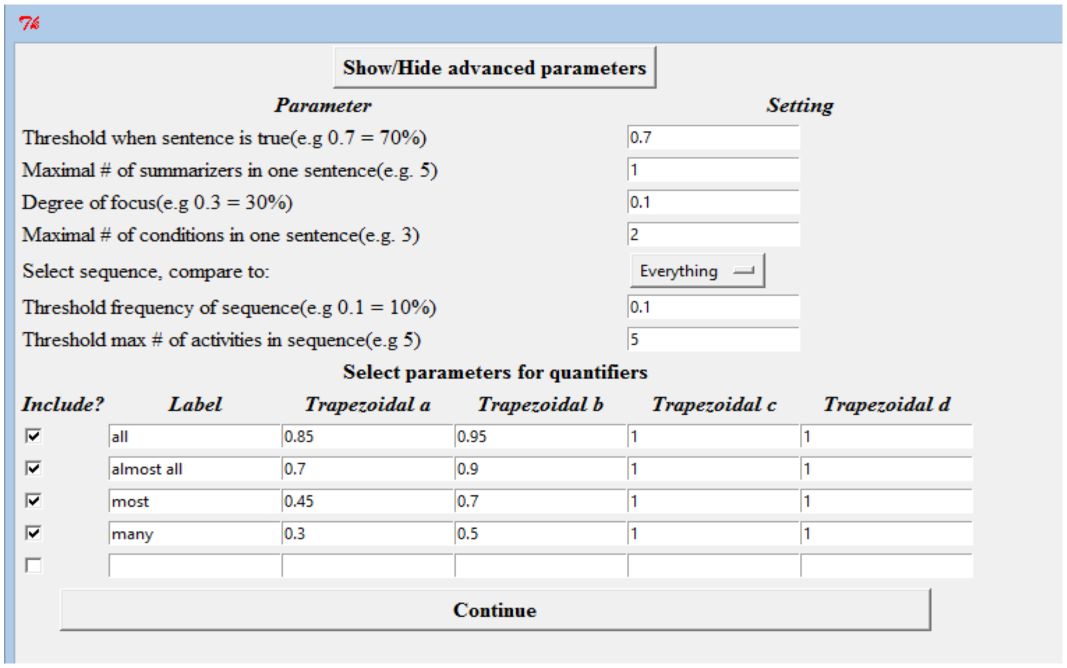


Parameter	Setting
Maximal # of summarizers in one sentence(e.g. 5)	1
Maximal # of conditions in one sentence(e.g. 3)	2

Select parameters for quantifiers

Include?	Label	Trapezoidal a	Trapezoidal b	Trapezoidal c	Trapezoidal d
☐	all	0.85	0.95	1	1
✓	almost all	0.7	0.9	1	1
✓	most	0.45	0.7	1	1
✓	many	0.3	0.5	1	1
☐					

Screen 18: Select parameters



Parameter	Setting
Threshold when sentence is true(e.g 0.7 = 70%)	0.7
Maximal # of summarizers in one sentence(e.g. 5)	1
Degree of focus(e.g 0.3 = 30%)	0.1
Maximal # of conditions in one sentence(e.g. 3)	2
Select sequence, compare to:	Everything
Threshold frequency of sequence(e.g 0.1 = 10%)	0.1
Threshold max # of activities in sequence(e.g 5)	5

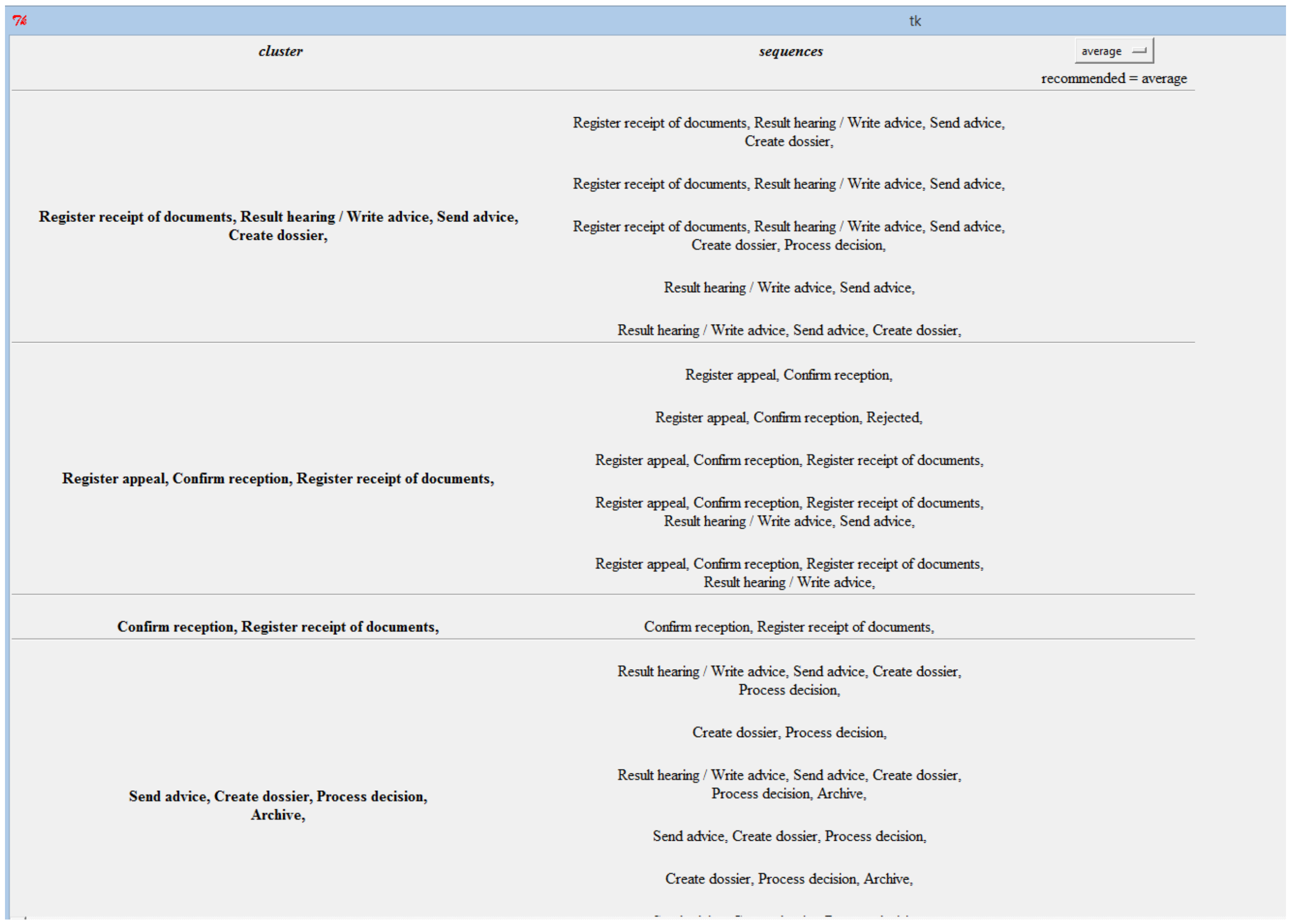
Select parameters for quantifiers

Include?	Label	Trapezoidal a	Trapezoidal b	Trapezoidal c	Trapezoidal d
✓	all	0.85	0.95	1	1
✓	almost all	0.7	0.9	1	1
✓	most	0.45	0.7	1	1
✓	many	0.3	0.5	1	1
☐					

Screen 19: Select parameters (advanced)

8) Select clustering

When sequences are analyzed, one can choose to cluster similar sequences. When sequences are clustered, one can create sentences about the complete group, instead of on specific sequences. The clustering of sequences is explained in more detail in Section 7.2. The screen related to the clusters is shown in Screen 20. In the first column, the representation of the clusters are shown. Every cluster contains one or multiple sequences, shown in the second column. When the user does not agree with the created clusters, another clustering method can be selected in the top right. The recommended clustering technique is expected to give the best results. If the user wants to focus on exact sequences, instead of on groups of sequences, the method 'None' can be selected, which creates separate clusters for all sequences.



recommended = average

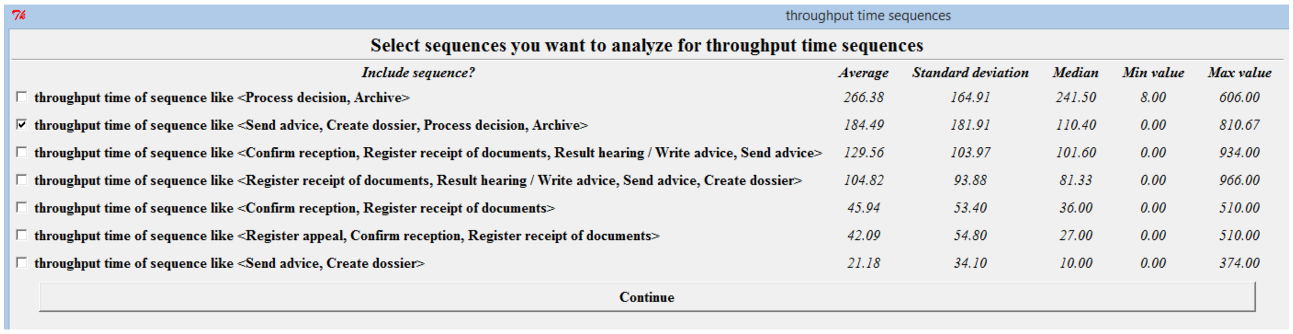
cluster	*sequences*
Register receipt of documents, Result hearing / Write advice, Send advice, Create dossier,	Register receipt of documents, Result hearing / Write advice, Send advice, Create dossier,
	Register receipt of documents, Result hearing / Write advice, Send advice,
	Register receipt of documents, Result hearing / Write advice, Send advice, Create dossier, Process decision,
	Result hearing / Write advice, Send advice,
	Result hearing / Write advice, Send advice, Create dossier,
Register appeal, Confirm reception, Register receipt of documents,	Register appeal, Confirm reception,
	Register appeal, Confirm reception, Rejected,
	Register appeal, Confirm reception, Register receipt of documents,
	Register appeal, Confirm reception, Register receipt of documents, Result hearing / Write advice, Send advice,
	Register appeal, Confirm reception, Register receipt of documents, Result hearing / Write advice,
Confirm reception, Register receipt of documents,	Confirm reception, Register receipt of documents,
Send advice, Create dossier, Process decision, Archive,	Result hearing / Write advice, Send advice, Create dossier, Process decision,
	Create dossier, Process decision,
	Result hearing / Write advice, Send advice, Create dossier, Process decision, Archive,
	Send advice, Create dossier, Process decision,
	Create dossier, Process decision, Archive,

Screen 20: Select clustering

9) Specify linguistic labels for times related to sequences

Depending on the features that have been selected in Step 5), features can be analyzed that focus on the chosen clusters. For some features, the membership functions have already been defined, e.g. whether a resource was present many times is based on the membership function defined for the feature: *# of times value xxx is selected for 'Resource', percentage based*. However, for some features, new membership functions have to be defined, e.g. the throughput time of a sequence. More information on this topic can be found in Section 7.4. Since the user probably does not want to analyze all sequences, a selection can be made, as shown in Screen 21. The selection can be based on these statistics. These statistics are shown in days.

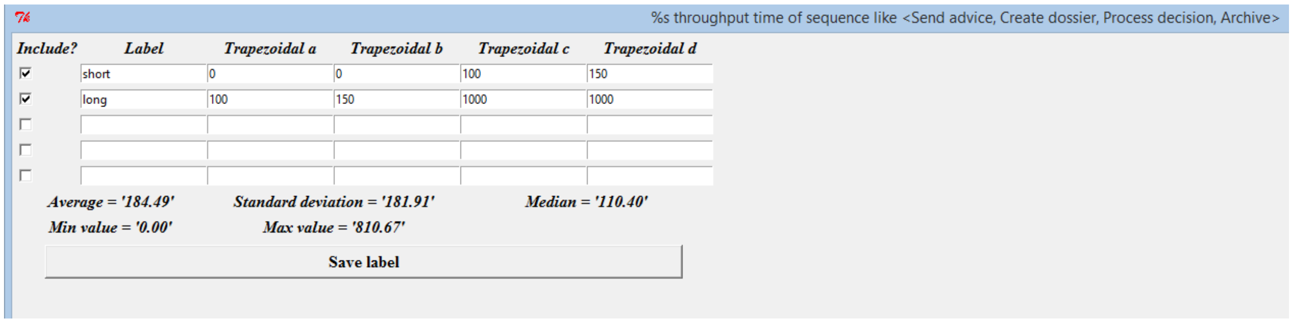
After the sequences are chosen that the user wants to analyze, membership functions have to be created. The user is asked whether the same membership functions are needed for all sequences, similar to Screen 14. Hereafter, membership functions are defined as is shown in Screen 22.


throughput time sequences

Select sequences you want to analyze for throughput time sequences

Include sequence?	Average	Standard deviation	Median	Min value	Max value
☐ throughput time of sequence like <Process decision, Archive>	266.38	164.91	241.50	8.00	606.00
☑ throughput time of sequence like <Send advice, Create dossier, Process decision, Archive>	184.49	181.91	110.40	0.00	810.67
☐ throughput time of sequence like <Confirm reception, Register receipt of documents, Result hearing / Write advice, Send advice>	129.56	103.97	101.60	0.00	934.00
☐ throughput time of sequence like <Register receipt of documents, Result hearing / Write advice, Send advice, Create dossier>	104.82	93.88	81.33	0.00	966.00
☐ throughput time of sequence like <Confirm reception, Register receipt of documents>	45.94	53.40	36.00	0.00	510.00
☐ throughput time of sequence like <Register appeal, Confirm reception, Register receipt of documents>	42.09	54.80	27.00	0.00	510.00
☐ throughput time of sequence like <Send advice, Create dossier>	21.18	34.10	10.00	0.00	374.00

Screen 21: Features related to times for sequence focused protoforms


%s throughput time of sequence like <Send advice, Create dossier, Process decision, Archive>

Include?	Label	Trapezoidal a	Trapezoidal b	Trapezoidal c	Trapezoidal d
☑	short	0	0	100	150
☑	long	100	150	1000	1000
☐					
☐					
☐					

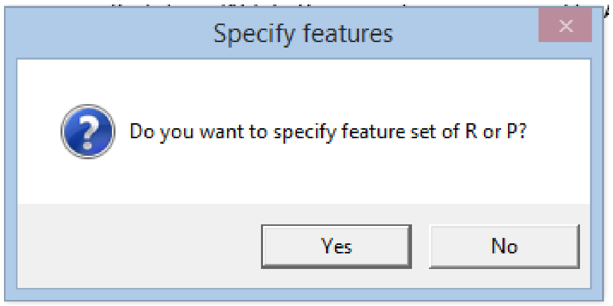
Average = '184.49' Standard deviation = '181.91' Median = '110.40'
Min value = '0.00' Max value = '810.67'

Screen 22: Specify linguistic labels for features related to times for sequence focused protoforms

10) Scope the analysis

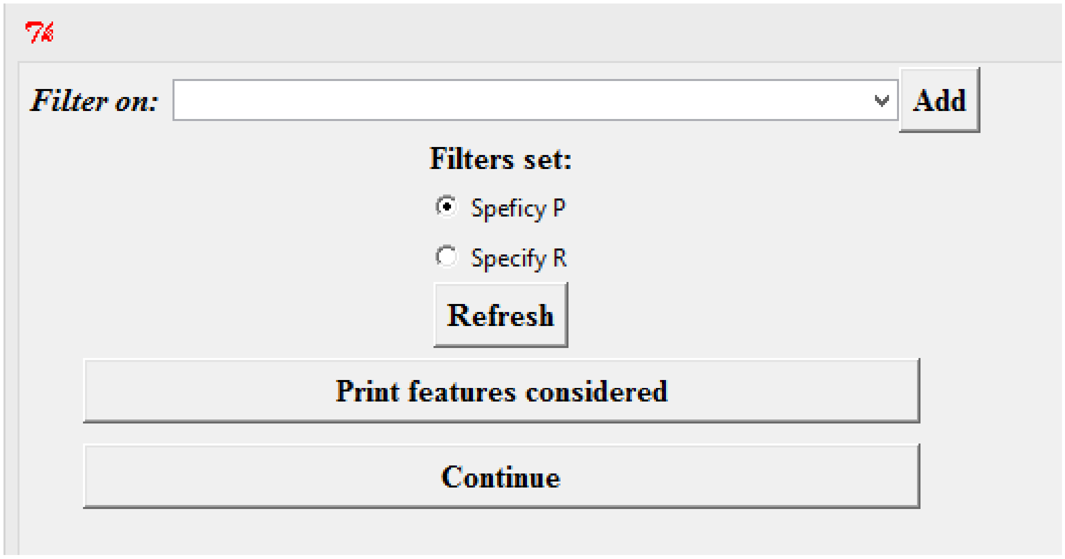
After all parameters are set, and all labels are defined, the user is presented Screen 23. If the user wants to focus on a specific feature, that is related to the complete case, for either the condition (R) or the summary (P), 'Yes' must be selected. If the user wants to look at all different summarizers and conditions, 'No' must be selected.

If 'No' is selected, the user proceeds to Step 11). However, if 'Yes' is selected, the screen shown in Screen 24 is presented. In the top row, a combobox is shown. When expanding the combobox, all features that can be analyzed are shown in alphabetical order, as shown in Screen 25. The user can either select a feature from the list, or enter a keyword, which is used to filter the values of the combobox, as shown in Screen 26.

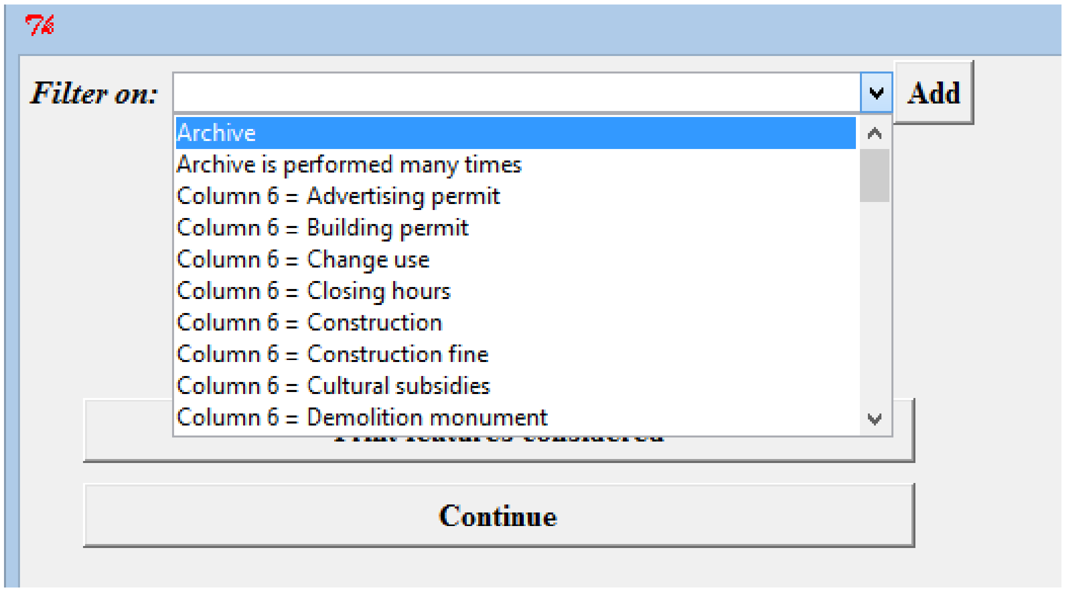

Specify features

Do you want to specify feature set of R or P?

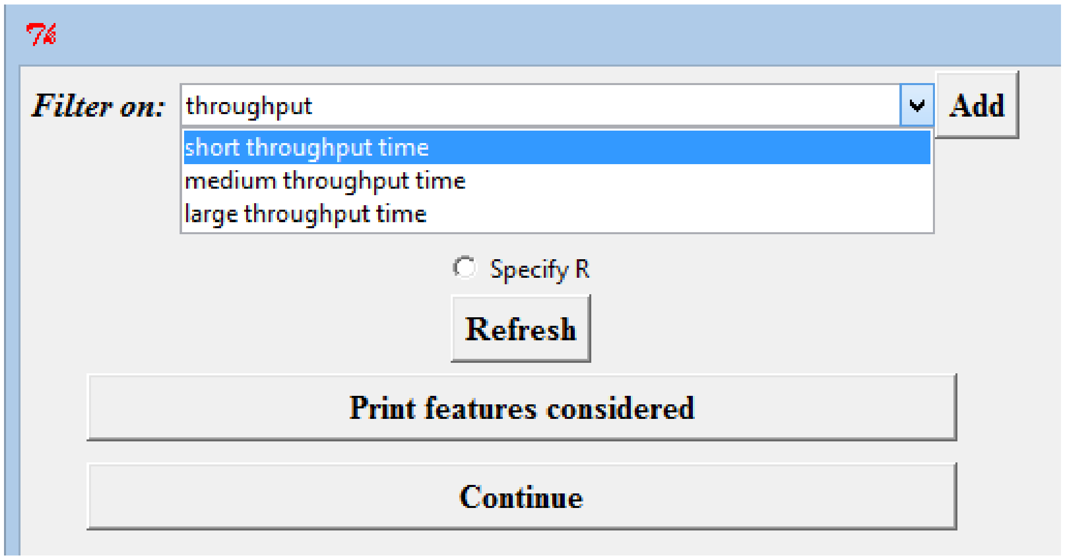
Screen 23: Question whether user wants to specify summarizer or condition



Screen 24: Screen to specify summarizer or condition


Filter on:
Add
Archive
Archive is performed many times
Column 6 = Advertising permit
Column 6 = Building permit
Column 6 = Change use
Column 6 = Closing hours
Column 6 = Construction
Column 6 = Construction fine
Column 6 = Cultural subsidies
Column 6 = Demolition monument
Continue

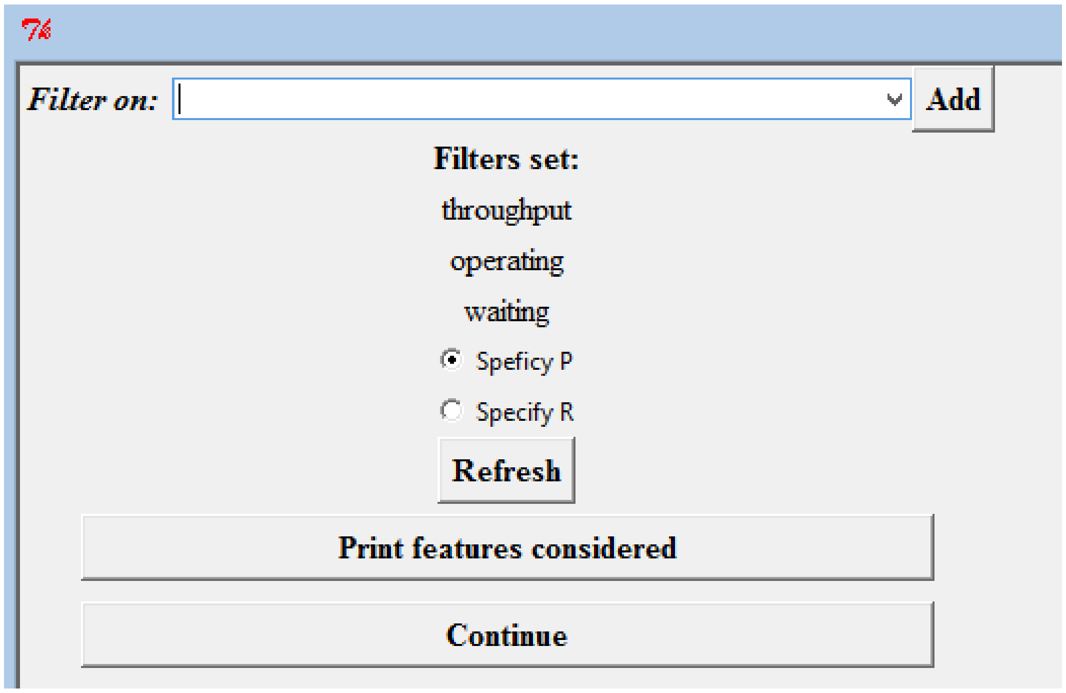
Screen 25: Combobox to help select features



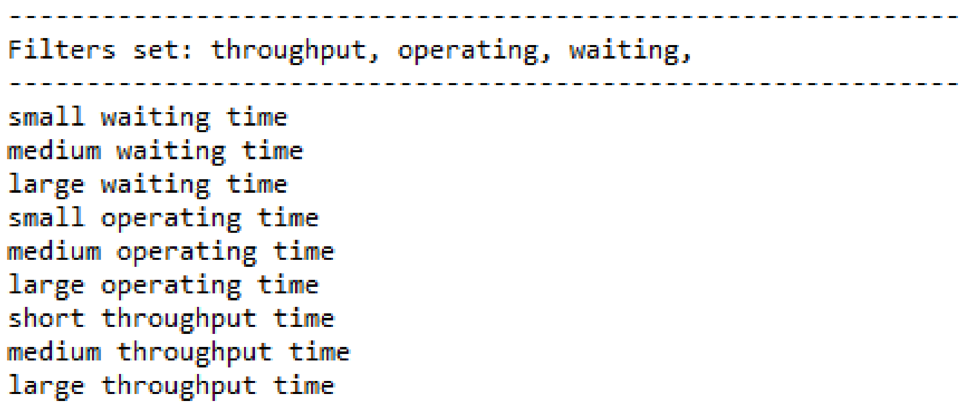
Screen 26: Combobox updates while entering text

By adding more filters, more features can be analyzed. In Screen 27, the filters ‘throughput’, ‘operating’ and ‘waiting’ have been set. This means that all features, that contain at least one of those filters, are analyzed. The button ‘Print features considered’ prints all features that are analyzed, using the filters set. An example is shown in Screen 28, which is based on the filters set in Screen 27. The button ‘Refresh’ can be used to clear the filters.

Note: Setting a scope can improve the performance of the tool significantly, since only a subset of the features has to be analyzed. Next to this, a lot of sentences, that are not in scope, are not shown, which makes the resulting set of sentences better readable.



Screen 27: Filters selected for summarizer



```
-------------------------------------------------------------
Filters set: throughput, operating, waiting,
-------------------------------------------------------------
small waiting time
medium waiting time
large waiting time
small operating time
medium operating time
large operating time
short throughput time
medium throughput time
large throughput time
```

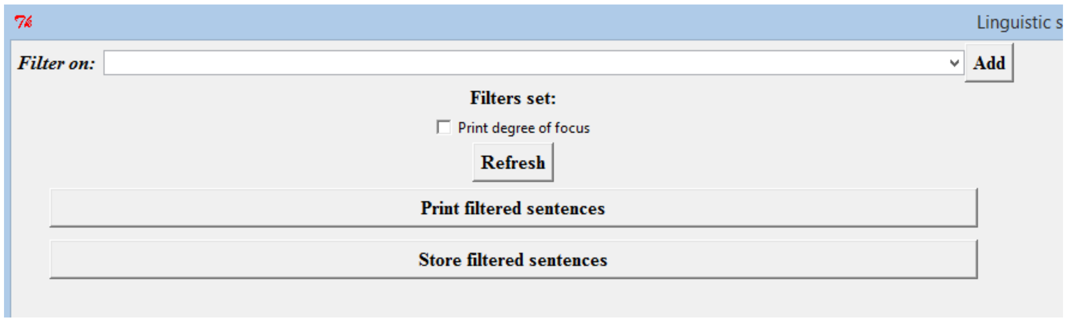
Screen 28: Resulting features that will be analyzed

11) Get results

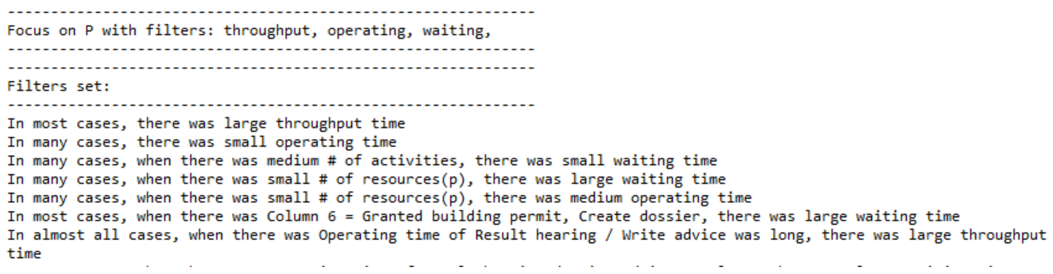
Finally, Screen 29 is shown, where the output can be stored and/or printed. In the top of the screen, filters can be set in the same way as shown in Step 10). The difference between this filter and the previously shown filter is, that this filter is based on conjunctions instead of on disjunctions, i.e. all filters have to be present in the resulting sentences.

One can choose to print the sentences in the command prompt, by use of the button ‘Print filtered sentences’, as shown in Screen 30. Another option would be to store the sentences in csv format, by use of the button ‘Store filtered sentences’. If this button is selected, the location is asked, where the file needs to be stored, similar to Screen 7. In both cases, the filters that are set are presented at the top of the result, as shown in, for example, Screen 30.

If the degree of focus needs to be shown, the option ‘Print degree of focus’ must be enabled. When the sentences are printed, using this option, the sentences are extended with an extra column, where the degree of focus is stored. For example, the sentence *“In most cases, there was a large throughput time”* contains no qualifier, and, therefore, the sentence is about all cases. However, the sentence *“In most cases, when there was medium # of activities, there was short waiting time”* is only about cases that have got *medium # of activities*. More information about the degree of focus can be found in Section 6.4.

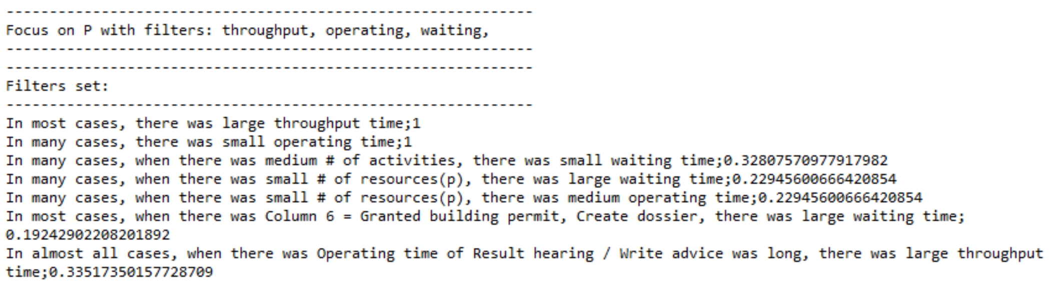


Screen 29: Screen for filtering, printing, and storing resulting set of sentences



```
------------------------------------------------------------
Focus on P with filters: throughput, operating, waiting,
------------------------------------------------------------
------------------------------------------------------------
Filters set:
------------------------------------------------------------
In most cases, there was large throughput time
In many cases, there was small operating time
In many cases, when there was medium # of activities, there was small waiting time
In many cases, when there was small # of resources(p), there was large waiting time
In many cases, when there was small # of resources(p), there was medium operating time
In most cases, when there was Column 6 = Granted building permit, Create dossier, there was large waiting time
In almost all cases, when there was Operating time of Result hearing / Write advice was long, there was large throughput
time
```

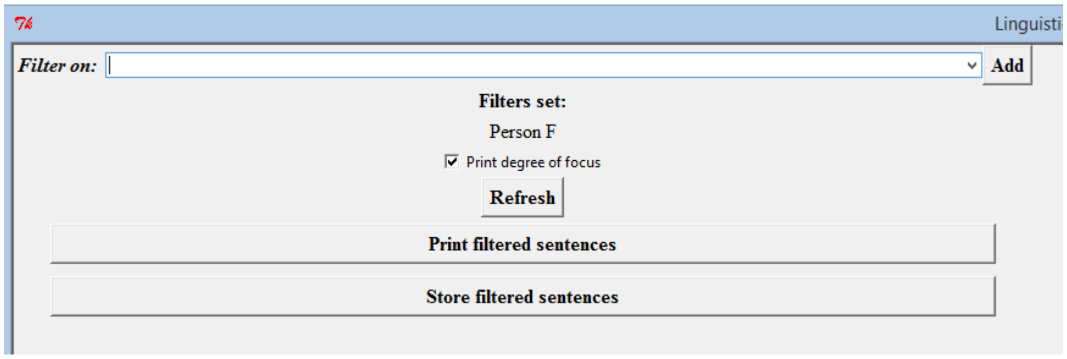
Screen 30: Resulting sentences with no filters on sentences



```
------------------------------------------------------------
Focus on P with filters: throughput, operating, waiting,
------------------------------------------------------------
------------------------------------------------------------
Filters set:
------------------------------------------------------------
In most cases, there was large throughput time;1
In many cases, there was small operating time;1
In many cases, when there was medium # of activities, there was small waiting time;0.32807570977917982
In many cases, when there was small # of resources(p), there was large waiting time;0.22945600666420854
In many cases, when there was small # of resources(p), there was medium operating time;0.22945600666420854
In most cases, when there was Column 6 = Granted building permit, Create dossier, there was large waiting time;
0.19242902208201892
In almost all cases, when there was Operating time of Result hearing / Write advice was long, there was large throughput
time;0.33517350157728709
```

Screen 31: Resulting sentences with no filters on sentences where degree of focus is turned on

An example of a filtering is shown in Screen 32, which specifies that all sentences must contain *Person F*. Example results, when this filtering is used, are shown in Screen 33. More filters can be added, to scope the set of sentences even more.



Linguisti

Filter on: Add

Filters set:

Person F

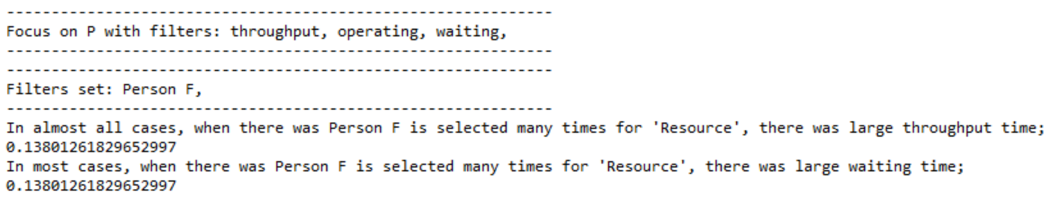
☑ Print degree of focus

Refresh

Print filtered sentences

Store filtered sentences

Screen 32: Filter set for resulting sentences



```
------------------------------------------------------------
Focus on P with filters: throughput, operating, waiting,
------------------------------------------------------------
------------------------------------------------------------
Filters set: Person F,
------------------------------------------------------------
In almost all cases, when there was Person F is selected many times for 'Resource', there was large throughput time;
0.13801261829652997
In most cases, when there was Person F is selected many times for 'Resource', there was large waiting time;
0.13801261829652997
```

Screen 33: Resulting sentences with filter 'Person F'

Appendix B: protoforms and related articles

Table 15: Protoforms and their related articles

Protoform Identifier	Protoform	Based on
(6)	$In\ Q\ cases, there\ was\ P$	[22], [23] and [24]
(7)	$In\ Q\ cases, when\ condition\ R\ was\ fulfilled, there\ was\ P$	[22], [23] and [24]
(8)	$Q\ cases\ contain\ a\ sequence\ like\ ABC$	[24]
(9)	$Q\ cases, when\ condition\ R\ was\ fulfilled, contain\ a\ sequence\ like\ ABC$	[24]
(10)	$In\ Q\ cases, that\ contain\ a\ sequence\ like\ ABC, there\ was\ P$	(7) and (9)
(11)	$Q\ cases, that\ contain\ a\ sequence\ like\ ABC, contain\ sequence\ like\ XYZ$	(7) and (9)
(12)	$In\ Q\ cases, there\ was\ P_S\ a\ sequence\ like\ ABC$	[24]
(13)	$In\ Q\ cases, when\ condition\ R\ was\ fulfilled, there\ was\ P_S\ for\ a\ sequence\ like\ ABC$	[24]
(14)	$In\ Q\ cases, when\ there\ was\ R_S\ for\ a\ sequence\ like\ ABC, there\ was\ P$	(13)
(15)	$In\ Q\ cases, when\ there\ was\ R_S\ for\ a\ sequence\ like\ ABC, there\ was\ P_S\ for\ a\ sequence\ like\ XYZ$	(13)
(16)	$E_T\ among\ all\ y's, Q\ are\ P$	[38]
(17)	$E_T\ among\ all\ R\ y's, Q\ are\ P$	[38]
(18)	$The\ more\ X\ is\ F, the\ more\ Y\ is\ G$	[39] and [40]
(19)	$In\ Q\ cases, there\ was\ P\ in\ process\ X$	(6)
(20)	$In\ Q\ cases, where\ condition\ R\ was\ fulfilled, there\ was\ P\ in\ process\ X$	(7)
(21)	$Q\ cases\ contain\ a\ sequence\ A * B * C$	[22] and (8)
(22)	$In\ Q\ cases, there\ was\ P_S\ for\ sequence\ A * B * C$	[22] and (12)
(23)	$In\ Q\ cases, when\ condition\ R\ was\ fulfilled, there\ was\ P_S\ for\ sequence\ A * B * C$	[22] and (13)
(24)	$The\ more\ activities\ ABC\ are\ done\ F, the\ more\ Y\ is\ G$	[39] and [40]
(25)	$In\ Q\ cases, sequence\ ABC\ was\ cyclic$	(8)
(26)	$In\ Q\ cases, when\ condition\ R\ was\ fulfilled, sequence\ ABC\ was\ cyclic$	(9)
(27)	$In\ Q\ cases, when\ sequence\ ABC\ was\ cyclic, there\ was\ P$	(10)
(28)	$In\ Q_1\ cases, there\ was\ P_1, but\ for\ Q_2\ cases, there\ was\ P_2$	[41]
(29)	$Usually/Rarely, there\ is\ P$	[42]
(30)	$Usually/Rarely, when\ condition\ R\ was\ fulfilled, contain\ a\ sequence\ like\ ABC$	[42]
(31)	$M\ every\ p\ unit, the\ data\ take\ high\ values$	[43] and [44]

Appendix C: Protoforms in scope and their statistics

Table 16: Protoforms in scope and their related statistics

Protoform Identifier	Statistics
(6)	$T(In\ Q\ cases, there\ was\ P) = \mu_Q\left(\frac{1}{n}\sum_{i=1}^{n}\mu_P(case_i)\right)$
(7)	$T(In\ Q\ cases, when\ condition\ R\ was\ fulfilled, there\ was\ P) = \mu_Q\left(\frac{\sum_{i=1}^{n}\mu_P(case_i)\wedge\mu_R(case_i)}{\sum_{i=1}^{n}\mu_R(case_i)}\right)$ $d_f(In\ Q\ cases, when\ condition\ R\ was\ fulfilled, there\ was\ P) = \frac{1}{n}\sum_{i=1}^{n}\mu_R(case_i)$
(8)	$T(Q\ cases\ contain\ sequence\ like\ ABC) = \mu_Q\left(\frac{1}{n}\sum_{i=1}^{n}\mu_{ABC}(case_i)\right)$
(9)	$T(\text{In Q cases, when condition R was fulfilled, contain a sequence like ABC}) = \mu_Q\left(\frac{\sum_{i=1}^{n}\mu_{ABC}(case_i)\wedge\mu_R(case_i)}{\sum_{i=1}^{n}\mu_R(case_i)}\right)$ $d_f(\text{In Q cases, when condition R was fulfilled, contain a sequence like ABC}) = \frac{1}{n}\sum_{i=1}^{n}\mu_R(case_i)$
(10)	$T(\text{In Q cases, that contain a sequence like ABC, there was P}) = \mu_Q\left(\frac{\sum_{i=1}^{n}\mu_P(case_i)\wedge\mu_{ABC}(case_i)}{\sum_{i=1}^{n}\mu_{ABC}(case_i)}\right)$ $d_f(\text{In Q cases, that contain a sequence like ABC, there was P}) = \frac{1}{n}\sum_{i=1}^{n}\mu_{ABC}(case_i)$
(11)	$T(\text{In Q cases, that contain a sequence like ABC, contain sequence like XYZ}) = \mu_Q\left(\frac{\sum_{i=1}^{n}\mu_{XYZ}(case_i)\wedge\mu_{ABC}(case_i)}{\sum_{i=1}^{n}\mu_{ABC}(case_i)}\right)$ $d_f(\text{In Q cases, that contain a sequence like ABC, contain sequence like XYZ}) = \frac{1}{n}\sum_{i=1}^{n}\mu_{ABC}(case_i)$
(12)	$T(In\ Q\ \text{cases, there was } P_S \text{ for a sequence like ABC}) = \mu_Q\left(\frac{\sum_{i=1}^{n}\mu_P(seq(ABC)_i)\wedge\mu_{ABC}(case_i)}{\sum_{i=1}^{n}\mu_{ABC}(case_i)}\right)$ $d_{sf}(In\ Q\ \text{cases, there was } P_S \text{ for a sequence like ABC}) = \frac{1}{n}\sum_{i=1}^{n}\mu_{ABC}(case_i)$

Protoform Identifier	**Statistics**
(13)	$T\left(\begin{matrix}In\ Q \text{ cases, when condition R was fulfilled, there was } P_S \text{ for} \\ \text{a sequence like ABC}\end{matrix}\right) = \mu_Q\left(\frac{\sum_{i=1}^{n}\mu_P(seq(ABC)_i)\wedge\mu_{ABC}(case_i)\wedge\mu_R(case_i)}{\sum_{i=1}^{n}\mu_{ABC}(case_i)\wedge\mu_R(case_i)}\right)$ $d_{sf}\left(\begin{matrix}In\ Q \text{ cases, when condition R was fulfilled, there was } P_S \text{ for} \\ \text{a sequence like ABC}\end{matrix}\right) = \frac{\sum_{i=1}^{n}\mu_{ABC}(case_i)\wedge\mu_R(case_i)}{\sum_{i=1}^{n}\mu_R(case_i)}$ $d_f\left(\begin{matrix}In\ Q \text{ cases, when condition R was fulfilled, there was } P_S \text{ for} \\ \text{a sequence like ABC}\end{matrix}\right) = \frac{1}{n}\sum_{i=1}^{n}\mu_R(case_i)$
(14)	$T(In\ Q \text{ cases, when there was } R_S \text{ for a sequence like ABC, there was P}) = \mu_Q\left(\frac{\sum_{i=1}^{n}\mu_P(case_i)\wedge\mu_{ABC}(case_i)\wedge\mu_R(seq(ABC)_i)}{\sum_{i=1}^{n}\mu_{ABC}(case_i)\wedge\mu_R(seq(ABC)_i)}\right)$ $d_{sf}(In\ Q \text{ cases, when there was } R_S \text{ for a sequence like ABC, there was P}) = \frac{1}{n}\sum_{i=1}^{n}\mu_{ABC}(case_i)$ $d_f(In\ Q \text{ cases, when there was } R_S \text{ for a sequence like ABC, there was P}) = \frac{\sum_{i=1}^{n}\mu_{ABC}(case_i)\wedge\mu_R(seq(ABC)_i)}{\sum_{i=1}^{n}\mu_{ABC}(case_i)}$
(15)	$T\left(\begin{matrix}In\ Q \text{ cases, when there was } R_S \text{ for a sequence like ABC,} \\ \text{there was } P_S \text{ for a sequence like ABC}\end{matrix}\right) = \mu_Q\left(\frac{\sum_{i=1}^{n}\mu_P(seq(XYZ)_i)\wedge\mu_{XYZ}(case_i)\wedge\mu_R(seq(ABC)_i)\wedge\mu_{ABC}(case_i)}{\sum_{i=1}^{n}\mu_{XYZ}(case_i)\wedge\mu_R(seq(ABC)_i)\mu_{ABC}(case_i)}\right)$ $d_{sf}\left(\begin{matrix}In\ Q \text{ cases, when there was } R_S \text{ for a sequence like ABC,} \\ \text{there was } P_S \text{ for a sequence like ABC}\end{matrix}\right) = \frac{\sum_{i=1}^{n}\mu_{XYZ}(case_i)\wedge\mu_R(seq(ABC)_i)\wedge\mu_{ABC}(case_i)}{\sum_{i=1}^{n}\wedge\mu_R(seq(ABC)_i)\wedge\mu_{ABC}(case_i)}$ $d_f\left(\begin{matrix}In\ Q \text{ cases, when there was } R_S \text{ for a sequence like ABC,} \\ \text{there was } P_S \text{ for a sequence like ABC}\end{matrix}\right) = \frac{\sum_{i=1}^{n}\mu_{ABC}(case_i)\wedge\mu_R(seq(ABC)_i)}{\sum_{i=1}^{n}\mu_{ABC}(case_i)}$
$\boldsymbol{\mu_Q}$ = membership of the quantifier $\boldsymbol{\mu_R}$ = the membership of the qualifier $\boldsymbol{\mu_P}$ = the membership of the summarizer $\boldsymbol{\mu_{ABC}}$ = the membership of sequence like ABC n the number of objects in the data ^ is the minimum seq_i(xxx) = the sequence under consideration(xxx) for the ith case	

Appendix D: Parameter relevance

Table 17: Relevant parameters per protoform

Protoform	Quantifiers	Minimal truth value	Maximum number of summarizers	Maximum number of qualifiers	Degree of focus	Maximum length of sequence	Threshold sequence occurrences	Comparison method
Protoform (6): In Q cases, there was P	✓	✓	✓	✗	✗	✗	✗	✗
Protoform (7): In Q cases, when condition R was fulfilled, there was P	✓	✓	✓	✓	✓	✗	✗	✗
Protoform (8): Q cases contain a sequence like ABC	✓	✓	Set to 1*	Set to 1*	✓	✓	✓	✓
Protoform (9): Q cases, when condition R was fulfilled, contain a sequence like ABC	✓	✓	Set to 1*	✓	✓	✓	✓	✓
Protoform (10): In Q cases, that contain a sequence like ABC, there was P	✓	✓	✓	Set to 1*	✓	✓	✓	✓
Protoform (11): Q cases, that contain a sequence like ABC, contain sequence like XYZ	✓	✓	Set to 1*	Set to 1*	✓	✓	✓	✓

Protoform	Quantifiers	Minimal truth value	Maximum number of summarizers	Maximum number of qualifiers	Degree of focus	Maximum length of sequence	Threshold sequence occurrences	Comparison method
Protoform (12): In Q cases, there was P_S for a sequence like ABC	✓	✓	Set to 1*	Set to 1*	✓	✓	✓	✓
Protoform (13): In Q cases, when condition R was fulfilled, there was P_S for a sequence like ABC	✓	✓	Set to 1*	✓	✓	✓	✓	✓
Protoform (14): In Q cases, when there was R_S for a sequence like ABC, there was P	✓	✓	✓	Set to 1*	✓	✓	✓	✓
Protoform (15): In Q cases, when there was R_S for a sequence like ABC, there was P_S for a sequence like XYZ	✓	✓	Set to 1*	Set to 1*	✓	✓	✓	✓

*For the protoforms related to sequences the *maximum number of summarizers* and/or the *maximum number of qualifiers* is set to one for the simplicity of the sentences.

Appendix E: Questionnaire

Table 18: questionaire evaluation based on TAM model [61] [59] [60] [62]

Variable	Question	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
Perceived Usefulness (PU)						
PU1	Using linguistic summarization in my job would enable me to accomplish tasks more quickly.					
PU2	Using linguistic summarization would improve my job performance.					
PU3	Using linguistic summarization in my job would increase my productivity.					
PU4	Using linguistic summarization would enhance my effectiveness on my job.					
PU5	Using linguistic summarization would make it easier to do my job.					
PU6	I would find linguistic summarization useful in my job.					
Perceived Ease of Use (PEOU)						
PEOU1	Learning to use linguistic summarization would be easy for me.					
PEOU2	I would find it easy to get linguistic summarization to do what I want it to do.					
PEOU3	My interaction with linguistic summary would be clear and understandable.					
PEOU4	I would find linguistic summarization to be flexible to interact with.					
PEOU5	It would be easy to become skillful at using linguistic summarization.					

Variable	Question	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
PEOU6	I would find linguistic summarization easy to use.					
	Intention To Use (ITU)					
ITU1	Assuming I have access to linguistic summarization, I intend to use it.					
ITU2	Given that I have access to linguistic summarization, I predict that I would use it.					
ITU3	I intend to find out more about linguistic summarization.					
ITU4	If I had material on linguistic summarization, I would study it.					
	Output Quality (OQ)					
OQ1	The quality of the output I get from linguistic summarization is high.					
OQ2	I have no problem with the quality of the system's output.					
	Result Demonstrability (RD)					
RD1	I have no difficulty telling others about the results of using linguistic summarization.					
RD2	I believe I could communicate to others the consequences of using the results generated by linguistic summarization.					
RD3	The results of using linguistic summarization are apparent to me.					
RD4	I would have difficulty explaining why using linguistic summarization may or may not be beneficial.					

Which tools (e.g. PROM or Disco) are you currently using to analyze processes?

____

What is your skill level using these tools?

- ☐ Beginner
- ☐ Intermediate
- ☐ Expert

Assuming I have access to linguistic summarization, and I have to analyze a business process, I will:

- ☐ not use linguistic summarization.
- ☐ use linguistic summarization in combination with tools I currently use.
- ☐ only use linguistic summarization.

I see the results of linguistic summarization as:

- ☐ noise.
- ☐ a subset of the results that tools I currently use provide.
- ☐ an extension to the results that tools I currently use provide.
- ☐ a standalone approach to analyze processes.

Appendix F: Disco models of appeals process

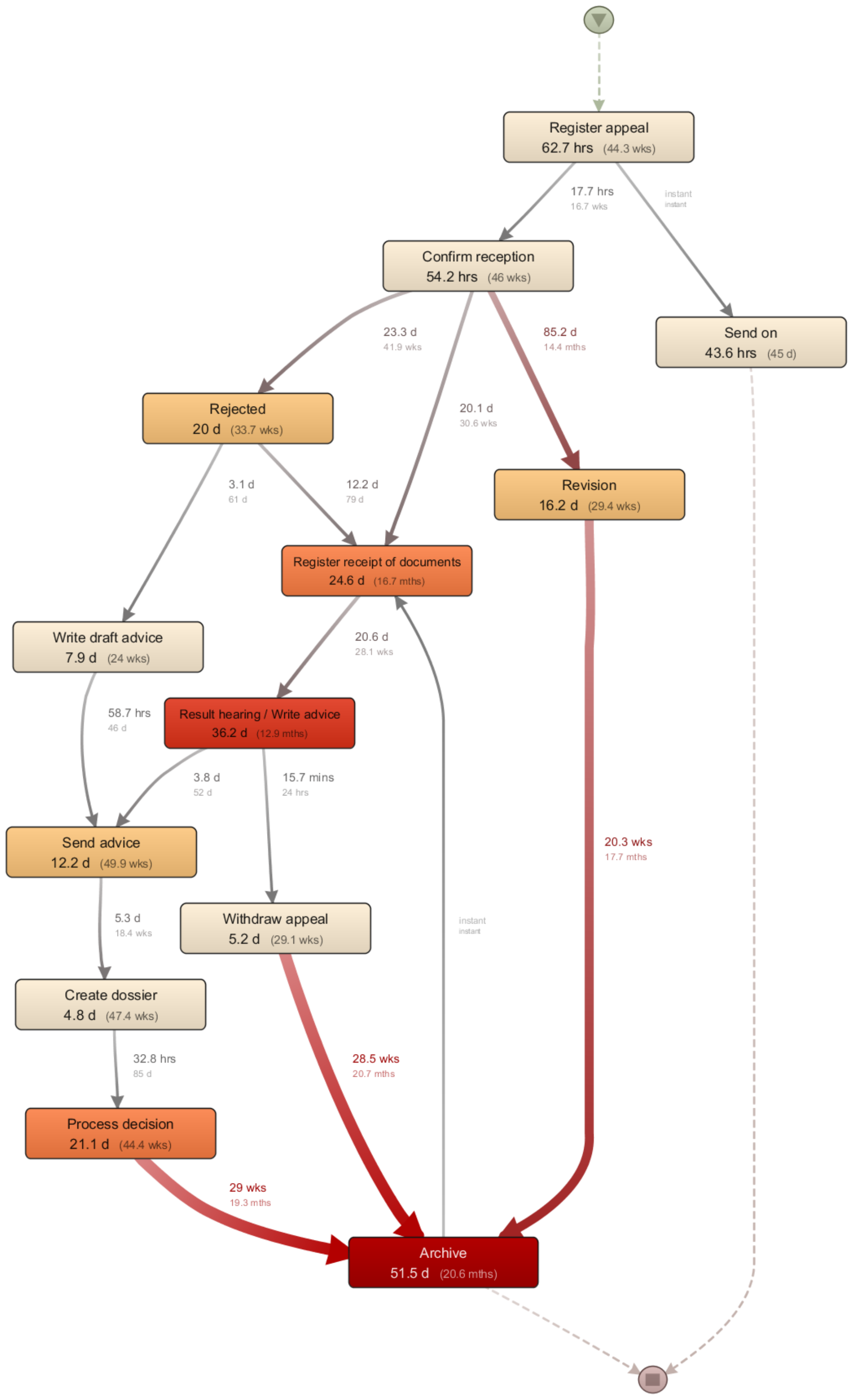


Figure 15: Average and maximal duration of appeal process