Goal-oriented Process Mining

Mahdi Ghasemi

Thesis submitted to the
Faculty of Engineering
in partial fulfillment of the requirements for the degree of

Ph.D. in Digital Transformation and Innovation

uOttawa

University of Ottawa
Ottawa, Ontario, Canada
December 2021

Abstract

Context: Process mining is an approach that exploits event logs to discover real processes executed in organizations, enabling them to (re)design and improve process models. Goal modelling, on the other hand, is a requirements engineering (RE) approach mainly used to analyze what-if situations and support decision making.

Problem: Common problems with process mining include the complexity of discovered "*spaghetti*" processes and a lack of *goal-process alignment*. Current process mining practices mainly focus on activities and do not benefit from considering stakeholder goals and requirements to manage complexity and alignment. The critical artifact that process mining practices rely on is the *event log*. However, using a raw version of real-life event logs will typically result in process models being too complex, unstructured, difficult to understand and, above all, not aligned with stakeholders' goals.

Method: Involving goal-related factors can augment the precision and interpretability of mined models and help discover better opportunities to satisfy stakeholders. This thesis proposes three algorithms for goal-oriented process enhancement and discovery (GoPED) that show synergetic effects achievable by combining process mining and goal-oriented modelling. With GoPED, *good* historical experiences will be found within the event log to be used as a basis for inferring *good* process models, and *bad* experiences will be found to discover models to avoid. The *goodness* is defined in terms of alignment with regards to three categories of goal-related criteria:

- Case perspective: satisfaction of individual cases (e.g., patient, costumer) in terms of some goals;
- Goal perspective: overall satisfaction of some goals (e.g., to decrease waiting time) rather than individual cases; and
- Organization perspective: a comprehensive satisfaction level for all goals over all cases.

GoPED first adds goal-related attributes to conventional event characteristics (case identifier, activities, and timestamps), selects a subset of cases concerning goal-related criteria, and finally discovers a process model from that subset. For each criterion, an algorithm is developed to enable selecting the best subset of cases where the criterion holds. The resulting process models are expected to reproduce the desired level of satisfaction. The three GoPED algorithms were implemented in a Python tool. In addition, three other tools were implemented to complete a line of actions whose input is a raw event log and output is a subset of the event log selected with respect to the goal-related criteria. GoPED was used on real healthcare event logs (an illustrative example and a case study) to discover processes, and the performance of the tools was also assessed.

Results: The performance of the GoPED toolset for various sizes and configurations of event logs was assessed through extensive experiments. The results show that the three GoPED algorithms are practical and scalable for application to event logs with realistic sizes and types of configurations.

The GoPED method was also applied to the discovery of processes from the raw event log of the trajectories of patients with sepsis in a Dutch hospital, from their registration in the emergency room until their discharge. Although the raw data does not explicitly include goal-related information, some reasonable goals were derived from the data and a related research paper in consultation with a healthcare expert. The method was applied, and the resulting models were i) substantially simpler than the model discovered from the whole event log, ii) free from the drawbacks that using the whole event log causes, and iii) aligned with the predefined goals.

Conclusion: GoPED demonstrates the benefits of exploiting goal modelling capabilities to enhance event logs and select a subset of events to discover goal-aligned and simplified process models. The resulting process model can also be compared to a model discovered from the original event log to reveal new insights about the ability of different forms of process models to satisfy the stakeholders' goals. Learning from good behaviours that satisfy goals and detecting bad behaviours that hurt them is an opportunity to redesign models, so they are simpler, better aligned with goals, and free from drawbacks that using the whole event log may cause.

Dedication

To the families of three uOttawa students whose flight PS752 never landed in Canada so that they could finish their research and study programs:

ALMA OLADI

MEHRABAN BADIEI ARDESTANI

SAEED KADKHODAZADEH KASHANI

And to:

HAMED ESMAEILION, who lost his family in that flight and never stopped the pursuit of truth and justice.

Acknowledgment

A Ph.D. is a long and arduous journey that cannot be successfully completed without help, support, and encouragement from many individuals. It would be impossible to list all those who have supported me throughout these years. I, therefore, wish to thank everyone who helped me to get to this point.

I would like to express my deepest gratitude to my supervisor, Prof. Daniel Amyot, whose support, guidance, and encouragement helped me throughout the years of my Ph.D. I would not have been able to complete this dissertation without his unswerving support and invaluable help. I will always be grateful to him for his *inspiring* method of supervision as well as his kindness, flexibility, and understanding, especially during the two years of my research conducted in the hard time of COVID-19. Merci beaucoup Daniel. J'apprécie votre aide et votre gentillesse.

Words fail to express how grateful I am to my parents for all the sacrifices they have made for me.

I want to express my deepest appreciation to my lovely wife, my best friend, and the love of my life, Rozita. Without your endless patience and great understanding, this work would not have been accomplished. Thank you for always standing by my side.

I would also like to thank my two lovely daughters, Hannah and Helia, who had to play alone many times as their dad had to work on the thesis!

I would also like to sincerely thank my committee members for generously providing valuable feedback and insightful comments during the comprehensive exam, the proposal defence, and the thesis defence. Special thanks go to the examiner committee – Prof. John Mylopoulos, Prof. Umar Ruhi, Prof. Liam Peyton, and Prof. Camille Salinesi – for reviewing my thesis and for their constructive feedback.

For my case study, I thank Dr. Randy Giffen who, as a medical expert and software engineer, kindly advised me about the event log I used and guided me on how to use this data for my work. I really appreciate his help and support.

Last but not least, I would like to express my genuine appreciation to my group-mates, who made a friendly and pleasant environment for me during these years.

This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), Ontario Graduate Scholarship (OGS), and the University of Ottawa.

Table of Contents

List of Figures

List of Models

List of Tables

List of Acronyms

Acronym	Definition
ACS	Academic Search Complete
ANN	Artificial Neural Networks
BDI	Belief-Desire-Intention
BI	Business Intelligent
BPI	Best Practice Implementation Indicator
BPM	Business Process Management
BPMN	Business Process Model and Notation
BSC	Balanced Scorecard
BWM*	Baum-Welch Algorithm
CDF	Cumulative Distribution Function
CHEO	Children's Hospital of Eastern Ontario
CJM	Customer Journey Mapping
CP	Clinical Pathway
CRP	C-reactive Protein
CrowdRE	Crowd-based Requirements Engineering
DGD	Diagnosis of Gestational Diabetes
DSR	Design Science Research
EDI	Electronic Data Interchange
GoPED	Goal-oriented Process Enhancement and Discovery
GoCC	Goal-oriented Conformance Checking
GRL	Goal-oriented Requirement Language
HIS	Healthcare Information Systems
HMM	Hidden Markov Model
IMF	Inductive Miner—infrequent
KAOS	Keep All Objects Satisfied
KPI	Key Performance Indicator
LDA	Latent Dirichlet Allocation
LTL	Linear Temporal Logic
MMM	Map Miner Method
MV	Maximum Variance
PDF	Probability Density Function
PI	Performance Indicator
PPI	Process Performance Indicators
PMaaS	Process Mining as a Service
RE	Requirements Engineering
RQ	Research Question
SL	Satisfaction Level
SLR	Systematised Literature Review
SOM	Self-organizing Map
SVM	Support Vector Machines
URN	User Requirements Notation
VA	Viterbi Algorithm

Chapter 1. Introduction

In this thesis, a novel method, called *Goal-oriented Process Enhancement and Discovery* (GoPED) is provided. This method supports process mining activities by taking advantage of both event logs and measurable goals in producing simpler and goal-aligned process models.

This chapter discusses the problem context and motivation, as well as the research questions. The research methodology this thesis follows is also explained, and the contributions are highlighted.

1.1. Problem Context and Motivation

Actors perform activities within business processes to reach their goals. The activities are observed as they are recorded in event logs. These logs, resulting from the execution of processes, might be characterized by various attributes such as identifiers, timestamps, activity names, transaction types (start, complete), resources, and associated costs. These logs are the main source of every study that aims to take *real-world processes* into consideration. *Process mining* is a maturing discipline, built upon process model-driven approaches and data mining, which is concerned with exploiting event data [1]. Various methods, tools and algorithms developed within the process mining community enable the analysis of event logs primarily through discovering underlying processes and creating process models. Process mining techniques are classified into three categories: i) *Discovery*, where a model is being created using event logs; ii) *Conformance checking*, where the data generated from the model is compared with the actual data in event logs to discover differences between the model and reality; and iii) *Enhancement*, where the desired data is used to improve or/and extend an existing process model.

In the context of requirements engineering (RE), *goal modelling* is used to support heuristic, qualitative, or formal reasoning about stakeholder and system goals, and ulti-

mately what-if/trade-off analysis and decision making [2]. While process mining is an activity-oriented approach and therefore focuses on "*how*", "*what*", "*where*", "*who*", and especially "*when*" questions, goal-oriented modelling focuses mainly on answering "*why*" questions that are not considered by process mining [3].

Process mining techniques assume that the cases within the logs of a process pursue some similar goals and do not consider the specific goals of individual cases or their satisfaction levels. This situation threatens the rationality behind the discovered models on one hand, and results in unstructured "spaghetti-like" processes on the other hand. Although they reflect reality, unstructured processes cover many exceptions, which hurts model understanding.

The process mining literature suggests some strategies for dealing with unstructured discovered processes [1]. The main idea is to take into account the frequency of activities and carefully filter out infrequent activities traces from the log before discovery. For example, keeping the activities that occur in at least 20% of cases is a way to simplify the model. In contrast to filtering strategies that change the log, abstraction techniques such as fuzzy mining [1] can be directly applied to the resulting process graph.

Goal-oriented approaches offer a way to document intentions and rationales, leading modellers to consider opportunities that stakeholders look for and vulnerabilities that they try to avoid. Such approaches also support answering "*what*" questions in a way that helps identify capabilities, services, and architectures required to satisfy stakeholder goals [2].

Therefore, a goal-oriented approach combined to process mining activities looks promising not only to identify capabilities, services, and architectures, but also to document and exploit goals and rationales and leverage them to improve process models and their realization. Process models that are aligned with goals are expected to produce high levels of goal satisfaction.

In this thesis, goals and intentions behind business process will be considered. The objective in this work is to develop a goal-oriented process mining method concerned not only with the sequencing of process activities (as is conventionally the case in process mining), but also with process goals and indicators. This method will show how to use goal-oriented requirements engineering concepts to enrich process mining activities.

1.2. Research Question

According to the above explanation, this research aims to develop an approach in which the synergy between process mining and goal modelling can contribute to increase the alignment between goals (or requirements) and prescribed or observed process models. The research question (RQ) of interest here is:

> **RQ:** How can we enrich process mining activities such that they exploit both event logs and measurable goals (derived from goal models) in order to generate models aligned with goals?

In this research, three new algorithms to address the research question are developed, implemented, and evaluated.

It is noteworthy that a goal here is mainly an objective that one or multiple stakeholders aim to achieve. For example, for an insurance claim process, a goal could be "to satisfy the customer" or "to reduce the claim processing time". The goals considered are at the business level, and are not about process model configuration (e.g., "the process model should have less than 8 steps") or about the process mining activities themselves (e.g., "the mined process model shall be available within two weeks").

1.3. Research Methodology

This research was conducted by following the Design Science Research (DSR) methodology [4], with a special attention to the version for information systems introduced by Peffers *et al.* [5].

The entry point is problem-centred and exploits two systematic literature reviews. In the first review [6], the status of process mining in general (and in healthcare in particular) was assessed and it was concluded that process mining is a growing research topic that emerged in the last decade and that deserves to be better developed. The second review [3] focused on goal-oriented process mining. This review concluded that, although process mining and goal modelling are growing research topics with considerable amounts

of publications, there are only a few rare studies conducted at their intersection, which we refer to as "goal-oriented process mining".

Acknowledging this gap, the research objective was defined as the development of methods that take advantage of goal models and enable analysts to (re-)design more satisfactory process models. In particular, the thesis focuses on developing a method (GoPED) to reduce complexity and improve goal-process alignment based on process mining. This method aims to make a beneficial connection between event logs, as the main input of process mining activities, and goals (and their indicators), as the main concept behind goal modelling. Beyond this prescriptive method artefact, the thesis provides other prescriptive artefacts, including three algorithms, four tools, and one method instantiation.

The demonstration and evaluation activities of DSR are performed by 1) using the method on existing logs, 2) a controlled experiment focusing on scalability, 3) a case study from the healthcare domain, and 4 a comparison with related work.

The main activities of the DSR methodology, the way they were instantiated in this thesis, and the various iterations performed are further detailed in Chapter 3.

1.4. Contributions

A set of three alternative algorithms addressing our research question represent the main contribution of this research: *Goal-oriented Process Enhancement and Discovery* (GoPED). These new algorithms take goal-related attributes into account in two of the main activities of process mining, namely process discovery and process enhancement. The GoPED method, which makes use of these algorithms, aims to exploit both measurable goals and event logs to generate simpler models aligned with goals. An instance of the method applied to a healthcare case study is also provided. Such goal-oriented method hence promises to contribute towards improving the rationality of the process discovery activities while simplifying the resulting models.

A toolset composed of four complementary tools (*GoPED, TraceMaker, EnhancedLogMaker,* and *EventLogRefiner*) that automate the proposed GoPED method is another contribution of this thesis. The three algorithms of GoPED are automated by the *GoPED* tool, which builds on top of existing optimization software. Preprocessing the input event log and producing the resulting event log for consumption by existing process mining tools

is enabled by the three other tools. These tools are publicly accessible online at https://github.com/Mahdi-Ghasemi.

Two published extensive systematic literature reviews on process mining in general and on goal-oriented process mining in particular also contribute to describing the state of the art in these areas.

1.5. Publications

Seven papers have been published in relation to this thesis:

1. **M. Ghasemi** and D. Amyot, "Process mining in healthcare: a systematised literature review", *International Journal of Electronic Healthcare (IJEH)*, vol. 9, no. 1, pp. 60-88, 2016. [6]
2. **M. Ghasemi** and D. Amyot, "From event logs to goals: a literature review of goal-oriented process mining", *Requirements Engineering*, vol. 25, no. 1, pp. 67-93, 2020. [3]
3. **M. Ghasemi**, "Towards goal-oriented process mining", [Doctoral Symposium paper], in *Proceedings of the International Requirements Engineering Conference*, IEEE CS, 2018, pp. 484-489. [7]
4. **M. Ghasemi**, "What requirements engineering can learn from process mining", in *1st International Workshop on Learning from other Disciplines for Requirements Engineering (D4RE)*, IEEE CS, 2018, pp. 8-11. [8]
5. **M. Ghasemi** and D. Amyot, "Data preprocessing for goal-oriented process discovery", in *IEEE 27th International Requirements Engineering Conference Workshops (REW)*, 2019, pp. 200-206. [9]
6. **M. Ghasemi** and D. Amyot, "Goal-oriented process enhancement and discovery", in *Proceedings of the International conference on business process management*, BPM, 2019, pp. 102-118. [10]
7. D. Amyot, O. Akhigbe, M. Baslyman, S. Ghanavati, **M. Ghasemi**, J. Hassine, L. Lessard, G. Mussbacher, K. Shen, and E. Yu, "Combining goal modelling with business process modelling: two decades of experience with the User Requirements

Notation standard", *Enterprise Modelling and Information Systems Architectures Journal*, 36 pages, 2022 (accepted November 5, 2021). [11]

The first (journal) paper reports on a systematized literature review we conducted on process mining in general and its contributions to the healthcare domain in particular. The second (journal) paper is a systematic literature review that assesses the state of *goal-oriented process mining.* These two studies represent the essence of the next chapter.

The third paper, presented as a doctoral symposium paper at the Requirements Engineering 2018 (RE'18) conference, proposes two methods (including GoPED) and reports on the research method, the gap/problems being addressed, preliminary solutions, expected contributions, and main foreseen challenges of this research.

The fourth paper, presented at an RE'18 workshop, highlights how requirements engineering can benefit from process mining components such as execution logs, process discovery, and conformance techniques for requirements elicitation, prioritization, and validation.

The fifth paper, presented at an RE'19 workshop, considers the data preprocessing step of the GoPED method and introduces two tools I developed for generating the enhanced log, as the primary input of GoPED algorithms, from a raw event log.

The sixth paper, presented at the Business Process Management (BPM 2019) conference, introduces a first version of the three GoPED algorithms refined here as the main contribution of this thesis.

Finally, goal-oriented process mining was highlighted in the last paper, where nine co-authors report on nearly two decades of combined goal/process modelling with URN in different areas.

1.6. Thesis Outline

The structure of this thesis is as follows. After clarifying the context and introducing some relevant definitions in the current chapter, the state of the art derived from contributed literature reviews will be highlighted in Chapter 2. Chapter 3 explains how the Design Science Research methodology was followed to conduct this thesis research. Then, Chapter 4 describes the basic concepts and structure of the GoPED method proposed as a

solution to address the research questions. The details of three GoPED algorithms are described in Chapter 5, while Chapter 6 presents an illustrative example to demonstrate the algorithms and their results. The experiments aiming to verify the feasibility and scalability of the algorithms applied on the different event logs are reported in Chapter 7. Then, in Chapter 8, the GoPED method is applied on a real event log of a healthcare system, and the results are discussed. Chapter 9 further discusses relations to closely-related work as well as limitations and threats to validity. Finally, Chapter 10 concludes and identifies important future work items.

Chapter 2. Literature Review

This chapter provides an overview of the main concepts of the research: process mining (Section 2.1) and goal modelling (Section 2.2). Then, the literature review conducted for this study is presented in three main parts. The first part, covered in Sections 2.3 and 2.4, is a systematic review of process mining in general (published in the *International Journal of Electronic Healthcare*, 2016 [6]). The second part, covered in Sections 2.5 and 2.6, is a review of goal-oriented process mining (published in *Requirements Engineering*, 2020 [3]). The reviews follow the systematic literature review guidelines of Kitchenham [12], with threats to validity discussed in Section 2.7. Finally, the third part (Section 2.8) provides a non-systematic overview of data preprocessing in process mining.

2.1. Overview of Process Mining

Process mining is an emerging discipline providing novel techniques to infer valuable knowledge and information from event logs, where logs typically include sequences of events (with timestamps and other attributes) from interleaving and identifiable process instances. For example, the electronic patient records in a healthcare system or the transaction logs of an enterprise resource planning system can be used to discover models describing processes, organizations, and products. Such event logs can also be used to compare reality with prior models to assess alignment [13].

Process mining is positioned as a powerful tool within a broader Business Process Management (BPM) context. Process mining can also be acknowledged as a new collection of Business Intelligent (BI) techniques with a time dimension. Notably, most BI tools focus on some querying and reporting together with visualization techniques and are not really "intelligent" enough to convey some process mining capabilities [14].

2.1.1 Brief History of Process Mining

According to Aldin and Cesare [15], Cook [16] published the first academic thesis on process mining while he was working on discovering process models from event logs in the

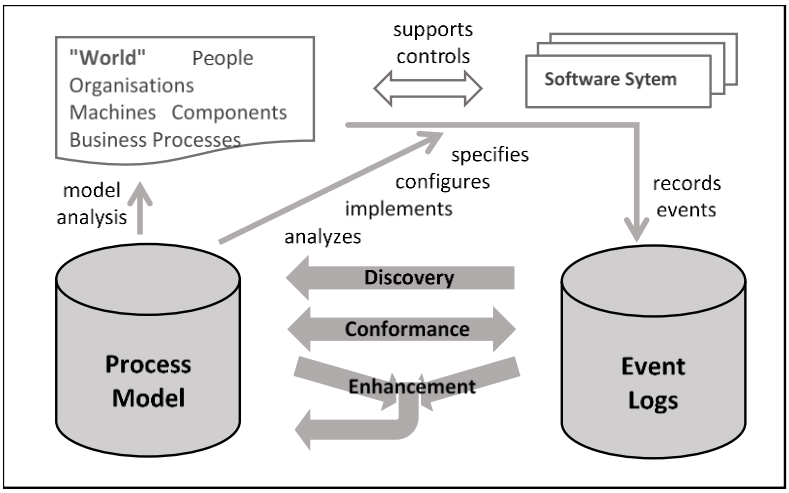


Figure 1. Three types of process mining: (1) discovery, (2) conformance, and (3) enhancement (inspired from van der Aalst [19])

context of software engineering. It was the first time that the term "*process discovery*" appeared. In 1998, Agrawal *et al.* [17], from IBM Almaden Research Center, introduced process mining in the business context and called it "*workflow mining*". Since then, several groups focused on process mining models and developed different algorithms and implemented tools and frameworks. Particularly, van der Aalst and Weijters [18] and their group from the Eindhoven University of Technology published the highest number of articles related to process mining.

2.1.2 Three Types of Process Mining

In general, process mining activities are categorised into three main types: i) Discovery, ii) Conformance, and iii) Enhancement (see Figure 1).

- **Discovery**: Here, the aim is to produce a process model, e.g., a Petri net [19], [20] or a BPMN model [21], using event logs. There is no prior model involved in a *process discovery* technique. The inferred model should be able to describe the observed behaviour of a process. The α-algorithm [48], which is able to discover a model based on sequences of events, is an example of discovery technique. Note that some models describing the data or organizational perspectives may be discovered [14]. Process discovery is acknowledged as the most prominent process mining technique.

- **Conformance**: In this type, an existing process model is compared with observed behaviour in the event logs [13]. Deviations between the model and the logs (e.g., activities in the model that do not exist in the log or vice versa) can be further analysed. Conformance checking can be used for business auditing and compliance checking. For example, replaying the event logs on top of the process model can highlight undesirable deviations and suggests fraud or inefficiencies. Conformance checking techniques can also be used for measuring the performance of process discovery algorithms and for repairing unaligned models.
- **Enhancement**: Whereas conformance checking assesses the alignment between reality and model, the third type of process mining aims to change or improve the *a priori* model. Note that it is supposed that a model, either discovered or produced manually, already exists [14], [22]. There are two types of enhancement activities: *repair*, where the model is modified to reflect reality better and bring the model closer to the reality, and *extension*, where a new perspective (time, costs, risks, resource usage, etc.) is added to the process model by cross-correlating it with the event logs. For example, one can identify bottlenecks by replaying timestamps in the event logs of a process model.

Figure 2 shows the inputs and outputs of the three types of process mining.

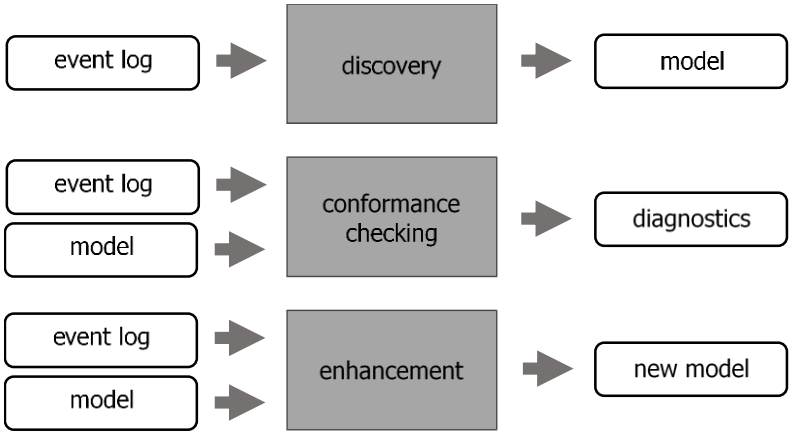


Figure 2. Three basic types of process mining, with inputs and outputs (inspired from [43])

2.1.3 Four Main Quality Dimensions of a Model

It is challenging to determine the quality of a process model discovered through a process mining algorithm. There are many dimensions that could be considered for such models. Four main quality dimensions are defined by van der Aalst [14] to assess the quality of a model: *fitness*, *simplicity*, *precision*, and *generalization*. A model is perfect in terms of fitness if all traces in the logs can be produced by the model from beginning to end. *Complexity* is defined by the number of nodes and edges used in the graph of the model. If the model allows for many other behaviours, even if there are no signs in the current logs that imply this behaviour, the model is under fitting and is its *precision* is insufficient. Finally**,** a model is not *general* enough if it explains the specific sample log, but another sample log of the same process produces a completely dissimilar process model. Process mining algorithms should manage the trade-offs between these four main dimensions.

2.1.4 Process Discovery Algorithms

There are many process mining algorithms proposed by researchers and practitioners. Each of these algorithms has different performances for different kinds of processes. As a result, it is difficult to select an appropriate process mining algorithm for a given application domain [23]. There are four typical characteristics of process discovery algorithms, with limitations further discussed by van der Aalst [14]:

- **Representational bias**: Most algorithms are not necessarily able to find all processes. There are some circumstances in event logs that cannot be covered by some algorithms. Some typical representational limitations forced by some process discovery algorithms are inabilities to represent concurrency, (arbitrary) loops, silent actions, duplicate actions, non-free-choice behaviour, OR-splits/joins, and hierarchy.
- **Ability to deal with noise**: Noisy behaviour, i.e., infrequent or exceptional behaviour, should not be considered in the discovered model, as users typically need to see the mainstream behaviour. Yet, it is hard to extract meaningful knowledge from activities or patterns that are extremely infrequent. Mature algorithms deal with this issue by abstracting from infrequent behaviour.
- **Completeness notion assumed**: Some algorithms work with the assumption that the event log includes all possible traces, i.e., a strong completeness assumption. This is

not realistic and results in overfit models. However, a weak completeness assumption tends to result in under-fit models.

- **Approach used**: Four typical families of approaches are: *direct algorithmic* (when the model is directly constructed from the event log), *two-phase* (when a "low-level model" is made first and then this model is converted into a "high-level model" that can show concurrency and more advanced patterns), *computational intelligence* (evolutionary approaches in which the model is constructed iteratively), and *partial* (similar to the discovery of association rules, but with respect to the order of events).

Some of the main process discovery algorithms include the following [14]:

- **α-algorithm**: This algorithm shows many of the general ideas used by many process mining algorithms and aids to understand the concept of process discovery [48]. This algorithm takes an event log as input and produces a Petri net showing the behaviour detailed in the log, including concurrency. However, the α-algorithm is not a very practical mining technique as it has some problems with noise, infrequent behaviour, incomplete behaviour, and complex routing constructs.
- **Heuristic Mining**: Heuristic mining algorithms, described by Weijters and Ribeiro [24], use graphs whose nodes represent activities and arcs represent causal dependencies. These algorithms consider *frequencies* of events and sequences when making a process model. The basic idea is that infrequent paths should not be accommodated into the model.
- **Genetic Process Mining**: This is an example of a process discovery algorithm that adopts a computational intelligence approach [25]. Like in any genetic algorithm, there are four main steps in this algorithm: initialization, selection, reproduction, and termination.
- **Region-Based Mining**: After converting an event log into a low-level transition system, one can synthesise a Petri net from it. In turn, this Petri net can be used to make a process model in terms of some other high-level notation (e.g., BPMN or UML activity diagrams). Folding a large transition system into a smaller Petri net by detecting concurrency is the challenge here. The main idea is to find regions that correspond to

places. A region is a set of states such that all activities in the transition system "agree" on the region [26].

Rozinat *et al.* [27] proposed an evaluation framework to compare the performance of algorithms, and to evaluate the validity of their results.

2.1.5 Process Mining Tools

van Dongen *et al.* [28], from the Eindhoven University of Technology, developed a "pluggable" environment for process mining: the *ProM framework* [29]. This framework is flexible in terms of the input and output formats, and allows reusing code throughout the implementation of novel process mining ideas. ProM uses MXML, SA-MXML [30], or XES [31] as input formats. Some import tools such as ProMimport and XESame are able to convert data from several sources into event logs. The ProM framework is well known among researchers and is the most popular and frequently utilised framework amongst process mining researchers [32].

Process mining capabilities are also provided by a growing number of commercial analysis tools such as *Disco* [33], *ARIS Process Performance Manager* [34], *Celonis Process Mining* [35], and *ProcessAnalyzer* [36]. These commercial tools embed many of the ideas developed in the context of ProM. More recent process mining tools are now also offered as services on the cloud [37], i.e., Process Mining as a Service (PMaaS).

2.1.6 Process Mining Manifesto

In 2012, the IEEE Task Force on Process Mining released the first version of the *Process Mining Manifesto* [38]. Fifty-three organizations and seventy-seven process mining experts supported and contributed to this manifesto. In this document, six guiding principles are defined to prevent users/analysts from making mistakes when applying process mining in real situations. Processes and information should be aligned in order to meet requirements related to compliance, efficiency, and customer service. In the manifesto, eleven important challenges are explained.

2.1.7 Process Mining: An Illustrative Example in Healthcare

One area that is amenable to process mining is healthcare [6]. *Healthcare Information Systems* (HIS) have numerous tables consisting of patient-related event data. Such data can be exploited to improve care processes while reducing costs [39]. Process mining techniques can play an important role in this regard, e.g., to improve compliance and performance. This section shows an example of process mining in healthcare.

To be able to improve processes, it is essential to understand what is really happening (*process discovery*) and analyse and interpret deviations from the expected process model (*conformance checking*). Furthermore, adding some perspective (times, costs, risks, resource usage, etc.) on top of the process model can help us identify and diagnose the roots of inefficiencies such as bottlenecks (*enhancement*).

For example, Table 1 shows event log of activities of 15 patients who have used the service of a health care system, namely the screening and diagnosis of gestational diabetes. Note that this table is not necessarily an original table that is stored in a database. It may be prepared from some raw tables that recorded all activities in order of the time that each event (activity) has happened.

If we use short names to encode the activities (e.g., a = admission), we obtain the traces shown in the third column of Table 1. The attributes of the various events, e.g., timestamp, resource, and data, are not shown in the table. There are some activities that are repeated for more than one patient; if we aggregate repeated ones, the event log can be shown as follows:

$$\mathbf{L = [\langle a, b, c, g\rangle^5, \langle a, b, c, d, e, c, g\rangle^5, \langle a, b, c, f, b, c, g\rangle^2,}$$
$$\mathbf{\langle a, b, c, f, b, c, d, e, c, g\rangle^2, \langle a, b, c, d, e, c, d, e, c, d, e, c, g\rangle^1]}$$

After analysing the events logs and the activities executed for patients, a process model such as the one shown in Figure 3 can be discovered. In this example, we use the α-algorithm [48] to obtain the control-flow model from event logs. This model can reproduce the traces observed and shown in Table 1.

The control-flow is represented in Figure 3 in terms of a Petri net, where transitions (squares) represent activities and places (circles) represent states. Transitions are connected through places. A transition is *enabled* and its corresponding activity can execute, if all input places hold a token (black dot). A token here essentially represents a patient.

Table 1. Event log example for the process of diagnosis gestational diabetes

Case	*Activities*	*Encoding*
Patient1	⟨admission, regular test, check the result, send the result⟩	**⟨a, b, c, g⟩**
Patient2	admission, regular test, check the result, request for advanced test, advanced test, check the result, send the result⟩	**⟨a, b, c, d, e, c, g⟩**
Patient3	⟨admission, regular test, check the result, request for repetition, regular test, check the result, send the result⟩	**⟨a, b, c, f, b, c, g⟩**
Patient4	⟨admission, regular test, check the result, request for repetition, regular test, check the result, request for advanced test, advanced test, check the result, send the result⟩	**⟨a, b, c, f, b, c, d, e, c, g⟩**
Patient5	⟨admission, regular test, check the result, request for advanced test, advanced test, check the result, request for advanced test, advanced test, check the result, request for advanced test, advanced test, check the result, send the result⟩	**⟨a, b, c, d, e, c, d, e, c, d, e, c, g⟩**
Patient6	⟨admission, regular test, check the result, request for advanced test, advanced test, check the result, send the result⟩	**⟨a, b, c, d, e, c, g⟩**
Patient7	⟨admission, regular test, check the result, request for advanced test, advanced test, check the result, send the result⟩	**⟨a, b, c, d, e, c, g⟩**
Patient8	⟨admission, regular test, check the result, send the result⟩	**⟨a, b, c, g⟩**
Patient9	⟨admission, regular test, check the result, request for repetition, regular test, check the result, request for advanced test, advanced test, check the result, send the result⟩	**⟨a, b, c, f, b, c, d, e, c, g⟩**
Patient10	⟨admission, regular test, check the result, request for advanced test, advanced test, check the result, send the result⟩	**⟨a, b, c, d, e, c, g⟩**
Patient11	⟨admission, regular test, check the result, request for repetition, regular test, check the result, send the result⟩	**⟨a, b, c, f, b, c, g⟩**
Patient12	⟨admission, regular test, check the result, send the result⟩	**⟨a, b, c, g⟩**
Patient13	⟨admission, regular test, check the result, request for advanced test, advanced test, check the result, send the result⟩	**⟨a, b, c, d, e, c, g⟩**
Patient14	⟨admission, regular test, check the result, send the result⟩	**⟨a, b, c, g⟩**
Patient15	⟨admission, regular test, check the result, send the result⟩	**⟨a, b, c, g⟩**

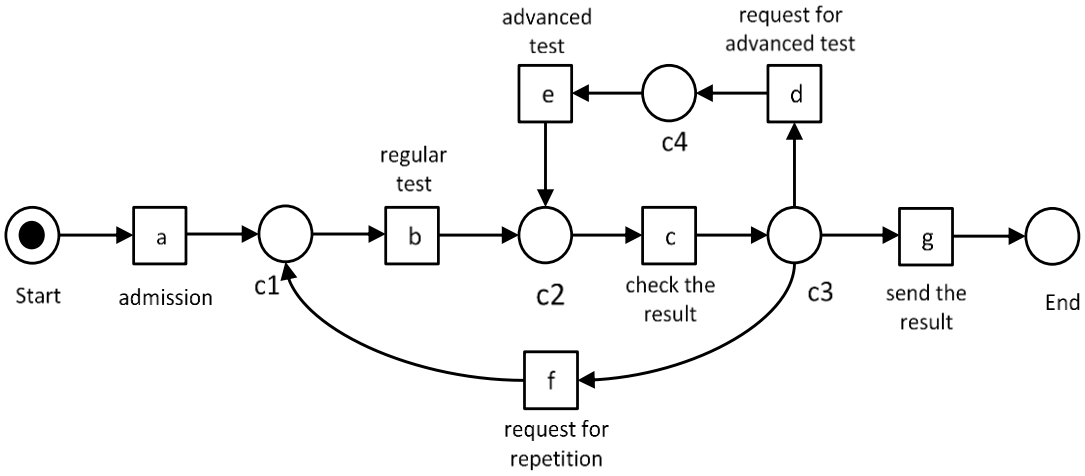


Figure 3. Process model discovered from an event log, using the α-algorithm

The process of screening and diagnosis of gestational diabetes starts by the admission of a pregnant patient. Transition admission has just one input place (start) and this place primarily has a token representing a patient. Hence, the corresponding activity is enabled and can happen. This is also called *firing*. When firing, the transition uses one token from each of its input places and produces one token for each of its output places. Hence, the firing of transition admission removes the token from input place start and produces a token for output place c1. Figure 3 shows the initial state (also called as *marking*), which consists of one token in place start. The marking after firing transition admission has a token in place c2. This goes on until the token gets to c3, where there is a choice between three activities, and possible iterations through other activities. The process ends with a token in place end.

There may be other information such as the resource (physician or nurse) executing the activity, the time of the events, or data that characterise the patient. By replaying the event log on top of the model shown in Figure 3, we can learn additional perspectives and enrich the model, as suggested in Figure 4 (some kind of conformance and enhancement).

As Figure 4 shows, according to Mans *et al.* [39], one can extend the process model with additional perspectives: the organizational perspective ("What are the roles and which resources are doing particular activities?"), the case perspective ("Which characteristics of a case impact on a particular decision?"), and the time perspective ("Where are the bottlenecks in the process?").

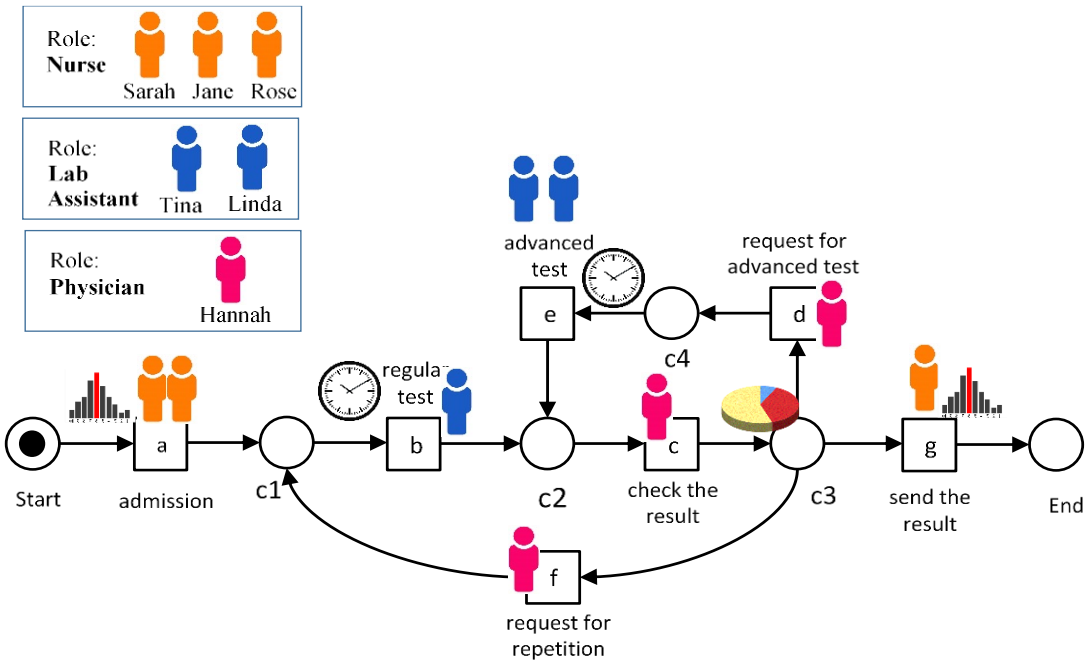


Figure 4. Process mining is also used to understand deviations, analyse bottlenecks, and monitor organizational behaviour

Analysis of the event log may disclose that Hannah is the only employee that performs the activities check the result, request for advanced test and request for repetition. This advises that there is a "physician role" and that only Hannah has this role. Activities regular test and advanced test are performed only by Tina and Linda, suggesting some "Lab Assistant" related to this activity, etc. Techniques for organizational process mining [14] will learn such organizational structures and relate activities to resources through roles. By taking advantage of resource information in the log, the organizational perspective can be added to the process model. Similarly, information on timestamps and frequencies can be used for adding performance related information to the model. Figure 4 shows that we can measure the time that passes between a request for advanced test (activity d) and the actual test (activity e). If this time is unusually long, process mining can identify the problem and discover possible root causes. If the event log has case-related information, this can be used to analyse the decision-making activities in the process. For instance, through this analysis, it may be learned that older patients require multiple operations [39].

2.2. Overview of Goal Modelling

Goals are objectives that stakeholders and systems aim to achieve. Goals can be used at different levels of abstraction. Goal-oriented requirements engineering is concerned with the use of goals for requirements elicitation, elaboration, structuring, specification, analysis, negotiation, documentation, and modification [40], [41]. Whereas processes excel at answering "*when*" questions, goals answer "*why*" questions that are not handled by process models, and hence both views are complementary.

2.2.1 Goal Model

A *goal model* is a directed graph that shows how goals contribute to each other [42]. AND-refinement links relate a goal to decomposing sub-goals. Here the parent goal is satisfied if and only if all sub-goals are satisfied. OR-refinement links relate a goal to some alternative refinements. This means that satisfying one of the sub-goals is sufficient for satisfying the parent goal. Contribution links between goals capture knowledge about the extent to which goals positively or negatively influence other goals. In this context, a conflict between two goals is present when the satisfaction of one goal may prevent the satisfaction of the other [41]. In case a goal contributes negatively to another goal in the goal model and the former one is satisfied, then the latter is unsatisfied. These kinds of rules are used for qualitative reasoning about goal satisfaction, e.g., in the Goal-oriented Requirement Language (GRL) [40].

In the context of requirements engineering, one benefit of goal modelling is to support heuristic, qualitative, or formal reasoning schemes. Moreover, one can verify that the requirements entail predefined goals [41]. Goals and requirements are elicited from multiple stakeholders with different viewpoints. Therefore, a wide range of conflicts can arise during analysis [43]. To reason about conflicting goals, one of the major benefits of goal modelling is to support trade-off analysis, targeting a solution that brings an appropriate balance when not all goals can be satisfied simultaneously. Goal models are reputed to be efficient in identifying and resolving conflicting concerns [43], [44] and in producing robust requirements through obstacle analysis [45], [46].

The goals' *intrinsic features* such as their types and attributes, together with their *links* to other goals and to other elements (e.g., containing actors, indicators, and requirements), make a goal model [41].

2.2.2 Functional and Non-Functional Goals

Several classification axes and types of goals have been proposed in the literature. *Functional* goals underlie services that should be delivered by the system. The functional goals characterize the expected system's behaviours, for example "Issue a receipt for every transaction". In contrast, *non-functional* goals talk about expected qualities of the system, for example "the receipt should be easy to read". *Non-functional* goals such as security, safety, usability, flexibility, customizability, and so forth, are often captured with *softgoals* [2].

2.2.3 Goal Modelling Languages

Several languages and notations have been developed to explicitly model goals and their relationships by the requirements engineering community [47]. For example, popular languages include Keep All Objects Satisfied (KAOS) [48], MAP [49] the NFR Framework [50], *i** [51], and Tropos [52]. More recently, GRL [40] has been recognized as an international standard for goal-oriented modelling, as part of a Recommendation of the International Telecommunication Union named *User Requirements Notation* (URN) [53]. URN is a standard notation capable of dealing with activity process models (in the form of Use Case Maps) *and* with goal models. URN also supports Key Performance Indicators (KPIs) in goal models and other concepts to measure and align processes and goals [54].

2.2.4 Goal Modelling: Two Illustrative Examples Using GRL and MAP

Another goal language, MAP, was proposed by Rolland *et al.* [49] to address a new category of intentional process models in the context of business requirements engineering. As MAP is the notation used in *intention mining* research activities (to be discussed in Section 2.6.2), it is important to explain that language here.

A Goal Modelling Example in GRL

GRL is a language that enables the analysis of a goal model by evaluating qualitative or quantitative satisfaction levels of the intentional elements (e.g., goals and tasks) and the actors constructing the model. There are different types of evaluations supported in GRL, including *quantitative* and *qualitative* evaluations, that exploit *contribution weights*, *importance values*, and *satisfaction values* in the model [55].

Evaluating the result of adapting different *strategies* on the intentional elements of the goal model is allowed by GRL. A *strategy* is defined by providing initial satisfaction values to some of the intentional elements. These values are then propagated to the other intentional elements through the different links that connect them. The effectiveness of the high-level requirements contained within the strategy can be determined through examination of the propagation results [55].

"What if?" questions are answered by *forward propagation*. Some intentional elements in the GRL model (source nodes, often leaves in the graph) are initialized within the strategy and propagated in a bottom-up way to higher-level intentional elements (targets) of the model. As shown in Figure 6-a, given the strategy that the student (with a partially good research expertise, and a fairly hot research topic) neither works nor applies for awards (and obviously does not get any scholarship) the propagation shows that the main goal of the student (to be graduated) would be partially denied (-38, on a scale from -100 to +100, where 0 is neutral). In addition, GRL supports analyzing the effect of a given strategy on the other actors or stakeholders. In the example, the aforementioned strategy results in that the student's family, in term of spending time with the student, is partially satisfied (+71). Also, as a result, another family's goal (having money to survive) would be fully denied (-100).

The overall satisfaction level of each actor (stakeholder) in the business can be also evaluated with GRL. The importance levels of the goals corresponding to a given actor and the satisfaction level of the goals make the overall level of satisfaction of the actor. In our example, although the student's family is happy about having enough time to spend with the student, the overall level of their satisfaction is -43 (partially denied). Here, we have some conflicting goals that are frequent during trade-off analysis. A given strategy can ended up with some conflicting goals and the main task here is to balance them.

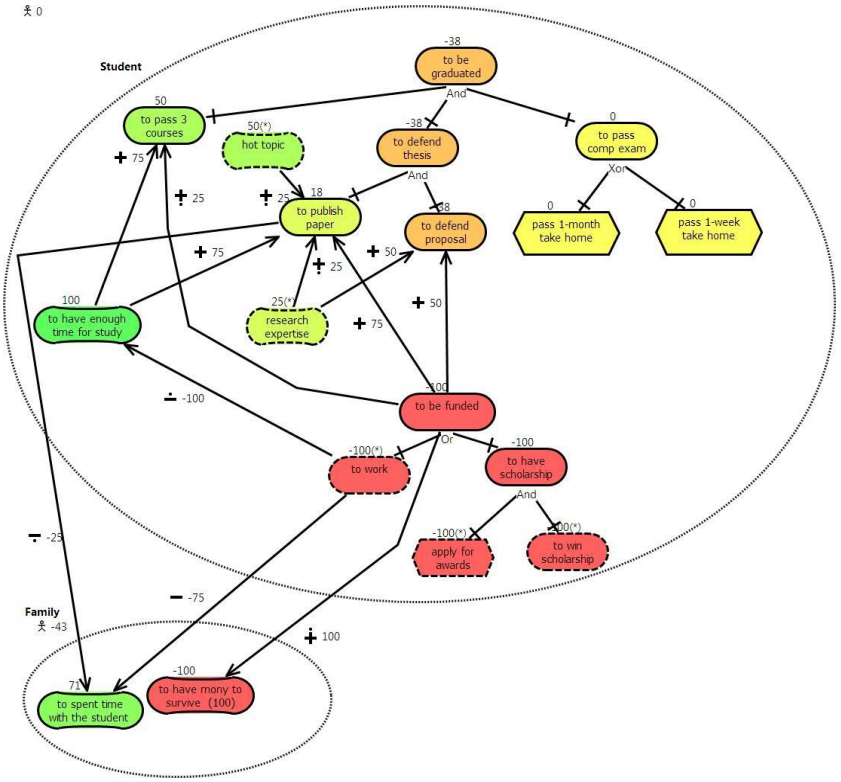


Figure 6-a. A GRL model with "*What if?*" analysis

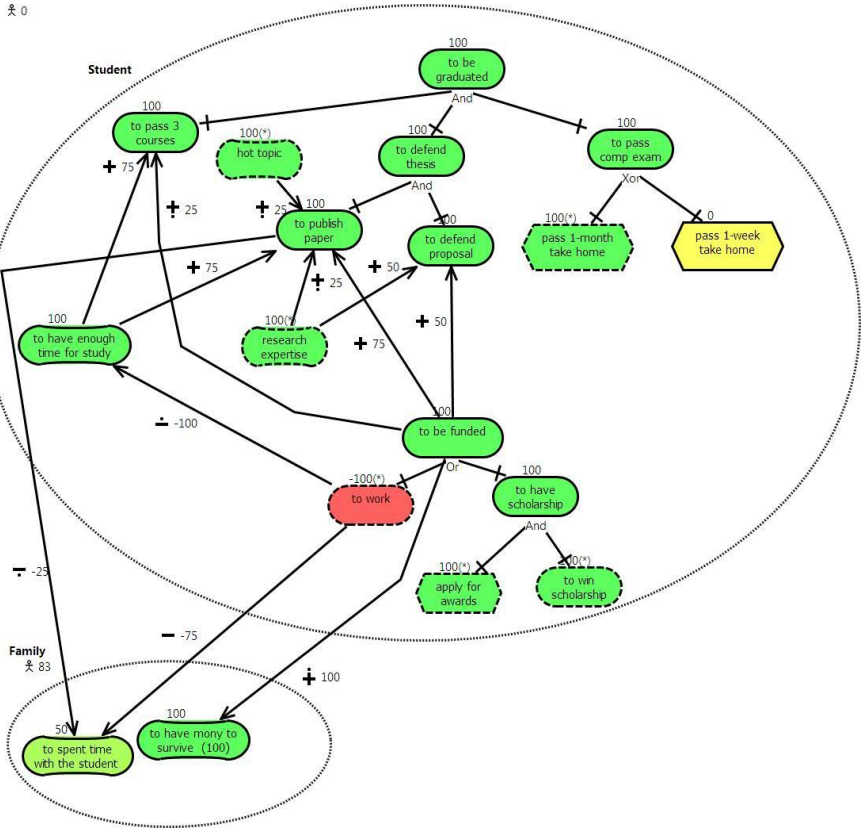


Figure 6-b. A GRL model with "*Is it possible?*" analysis

Also, desired evaluation values of main goals (often roots in the graph) can be propagated in a top-down manner to lower-level intentional elements (sources). *Is it possible?* questions are answered by such a propagation, which aims to find a set of alternatives that, when satisfied, lead to the initial values provided in top-level goals. In our example, if the main goal of the student is assumed as fully satisfied, the backward propagation shows that the goals "to pass 3 courses", "to defend thesis", and "to pass comp exam" must be fully satisfied (see Figure 6-b). Continuing over these three goals, we find that "to publish paper", "to defend proposal", "to have enough time for study", "to be funded", "research expertise", "hot topic" and one of the two tasks related to "to pass the comp exam" must be fully satisfied. Consequently, in order to fully satisfy "to be funded", one of two subgoals must be fully satisfied.

Full satisfaction of "to work" causes full denial of "to have enough time for study" and is not acceptable. Therefore, "to have scholarship" must be fully satisfied. Hence, the student should apply for awards and get them. In this situation, the overall satisfaction level for the family would be 83 (very satisfied, although not entirely).

This type of top-down analysis can also be turned into optimization problems that can be fed to commercial solvers, which then return the needed strategies, when they exist [56].

A Goal Modelling Example in MAP

A process, while performing, is not limited to linear activities; actors, based on their context, have a variety of choices for performing a task. MAP models direct the actor by proposing dynamic choices aligned with their intentions. A MAP process model is a labelled directed graph that consists of *intentions* as nodes and *strategies* as arcs between intentions. Note that MAP strategies are different from GRL strategies (the latter being a set of initial satisfaction values assigned to some intentional elements). The directed nature of the graph shows how the intentions can follow one another. Figure 7 shows a simple example of a MAP intentional model for buying a smart phone.

There are three key elements that make a MAP: *Intention*, *Strategy*, and *Section*. An *Intention* is a goal that can be achieved by performing some activities. For example, there are two intermediate intentions in Figure 7: Selection of a smart phone, and Buying

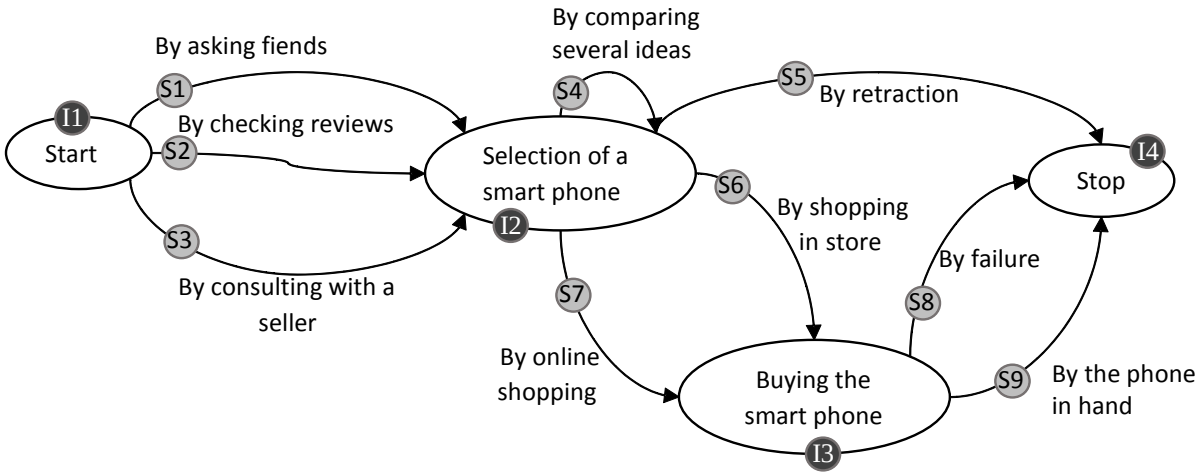


Figure 7. A MAP intentional model for guiding users who want to buy smart phones. There are four intentions (I1-I4, including Start and Stop) and nine strategies (S1-S9) available to achieve the intentions.

the smart phone. A *Strategy* is a means to achieve an intention. For example, "By asking friends" is a strategy to achieve the phone selection intention. Finally, a *Section*, is a triplet composed of a *Source Intention*, a *Target Intention*, and a *Strategy* linking them. Therefore, a MAP consists of a several sections, each of which being represented as a triplet $\langle I_s, I_t, S_{st} \rangle$. In our example, one such section is ⟨Selection of a smart phone, Buying the smart phone, By online shopping⟩.

In a MAP, there might be several alternative links from I_s to I_t, each corresponding to an explicit strategy (multi-thread flow) to reach I_t from I_s. There might also be several strategies from different intentions to reach an intention I_t (multi-flow paths to achieve an intention). Finally, a MAP can contain reflexive flows, i.e., I_s and I_t are the same intention. There are two special intentions called *Start* and *Stop*, which represent respectively the intentions to start crossing the map and to stop crossing it. As there can be several strategies that could be selected between two successive intentions, there are usually many paths in the graph from *Start* to *Stop*.

Rolland and Salinesi [57] have introduced and discussed goal/strategy MAPs as an example of goal models conceived to meet the challenges faced by *multi-purpose* systems, i.e., systems that incorporate variability in the functionality they provide and that can self-adapt to different situations. Kraiem *et al.* [58] proposed a novel approach for mapping intentional process models represented in MAP to business process models expressed in

the Business Process Model and Notation (BPMN) [21]. The proposed approach links the MAP elements to BPMN elements in order to align the intentional and operational levels.

2.3. Literature Review of Process Mining

In this section, we aim to review the papers related to process mining in healthcare. The trends of research on process mining in general and in healthcare are discussed. Important literature-review papers generally related to process mining are first briefly reviewed. Then, we narrow down the scope and focus on the healthcare domain.

Inspired by the work of Kitchenham *et al.* [59], we did our first systematised literature review (SLR1) to address the following three research questions:

- **RQ1_SLR1:** What is the trend of volume of research activities related to process mining and what is the share of healthcare in this trend?
- **RQ2_SLR1:** Are there any literature reviews specifically related to process mining? If so, how many of them are done in a systematic way and what are their main contributions? Is any of them related to care processes in particular?
- **RQ3_SLR1:** Do the literature reviews that are published give us a good understanding about process mining, especially in healthcare?

In this research, we considered four of the most relevant search engines in information technologies, namely Scopus, Google Scholar, IEEE Explore, and Academic Search Complete (ASC). In addition, we also considered PubMed (also covering Medline), Embase, Clinical Evidence, and the Cochrane Library, which are four healthcare indexes. The last four ones were included to cover the papers that are likely included only in healthcare databases. To conserve the ability of trend analysis, we consciously did not limit our review to a particular time period. However, very recent work is not included as this review was performed in 2016.

2.3.1 Trend of Volume of Research Activities

First, we ran an abstract query looking for papers related to process mining in general. We searched for "`process mining`" within the text of papers and, as shown in the first row of Table 2, Google Scholar and Scopus returned thousands, IEEE Explore returned about

one thousand, ASC less than three hundred, and PubMed and Embase returned less than 30 papers each. Two other search engines, Clinical Evidence and Cochrane Library, did not return any paper relevant to process mining. Based on these numbers, for the first four engines, we limited the search to metadata (title, abstract and keywords) in order to find manageable, meaningful, and relevant results. As Google Scholar is unable to search within abstracts and keywords, it was limited to only the title (patents and citations were also excluded). This limitation reduced the number of results dramatically and enabled us to analysis the trend of research activities in process mining while avoiding noise from less relevant papers. However, if one wants to sensitise the papers and review the results in details, the query needs to be further constrained. The result of the second query is shown in the second row of Table 2.

Table 2. Number of papers returned by the search engines

	Search within	*Scopus*	*Google Scholar*	*IEEE Explore*	*ASC*	*PubMed*	*Embase*
1	Full text	4,177	10,200	1,009	248	29	23
2	Limited	1,845	1,120	343	139	29	23
3	Extra limited	1,780	11,20	343	139	29	23
4	Removing duplications, independently within each search engine	1,744	1,011	342	139	28	21
5	Removing duplications, across all engines			***2,371***			

Reviewing the returned papers published before 2001 revealed that they are mostly related to processes in the chemical or mining industry. So, we added an extra constraint to the second query to exclude them: `"process mining" AND NOT ("mineral" OR "caving" OR "chemistry" OR "chemical")`, with impact only on Scopus; Google Scholar only searches within the titles, and the other engines already focus on domains other than the mineral industry.

Then, we focused on the results to find and eliminate duplicated papers in each search engine, in a trackable manner. The fourth row of Table 2 shows the numbers of

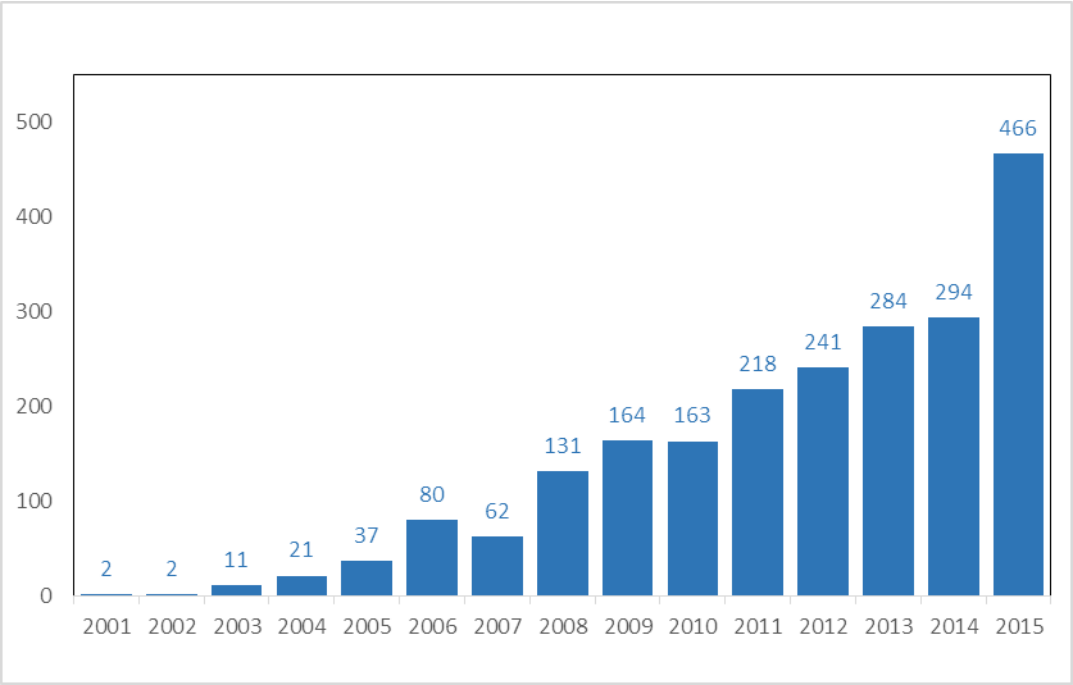


Figure 8. Number of process-mining-related publications, per year.

papers after removing this kind of duplication. Finally, we searched for duplicates across the results of the six search engines and after removing them, 2371 papers remained.

Based on the year of publication, the trend of the number of unique publications related to process mining is illustrated in Figure 8. According to this figure, process mining is a growing research topic that has emerged in the last decade, and recently it has received considerably higher research interest.

In order to find the share of the healthcare domain in process mining related research, we defined another query with additional constraints. This time, our query was `"process mining" AND (healthcare OR clinic OR hospital OR care OR clinical OR health)`. This query covered the essential keywords in the healthcare domain. The raw results of the search engines, where the query was limited to titles and keywords, are shown in Table 3. Given the specialised area of PubMed and Embase, it is not surprising to see that a majority of papers returned by these databases are related to healthcare.

Table 3. Process mining in healthcare – automatic search results

	Scopus	*Google Scholar*	*IEEE Explorer*	*ASC*	*PubMed*	*Embase*
Search Results	128	58	34	19	23	16

By comparing the 278 papers that the search engines have returned and after removing their duplicates, we get a total of 168 unique papers. A quick assessment of these papers confirmed that our customised queries have worked well as a majority of these papers are relevant.

To complete the answer to research question RQ1_SLR1, Figure 9 illustrates the number of publications related to process mining in healthcare and its growing trend. Moreover, the share of research papers related to healthcare in the whole process mining domain is shown in Figure 10. It can be seen from this latter figure that this share has a fairly increasing trend over the recent years.

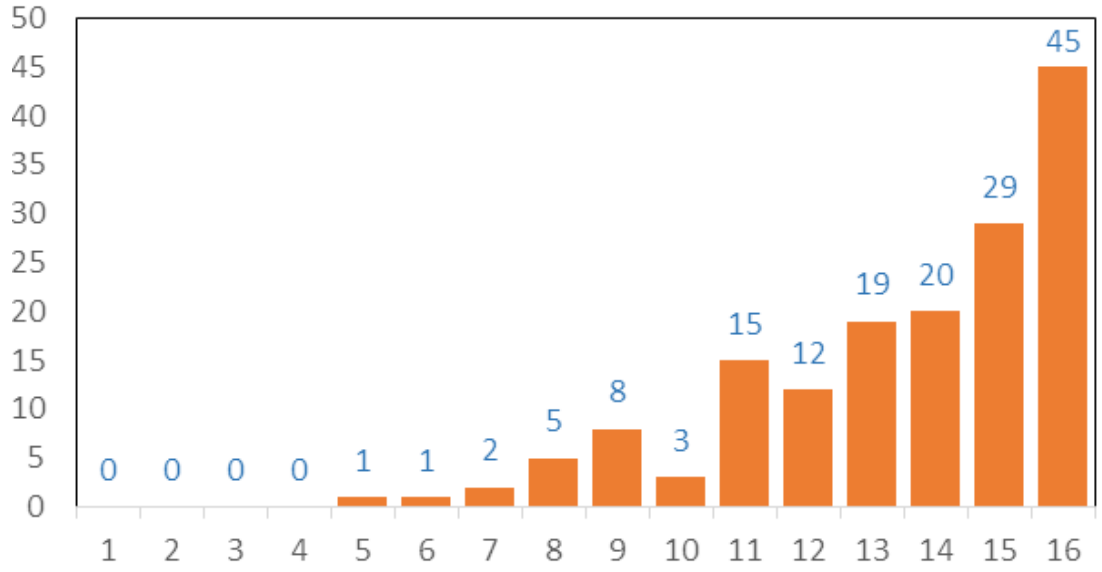


Figure 9. Number of publications related to process mining in healthcare, per year (2000-2015)

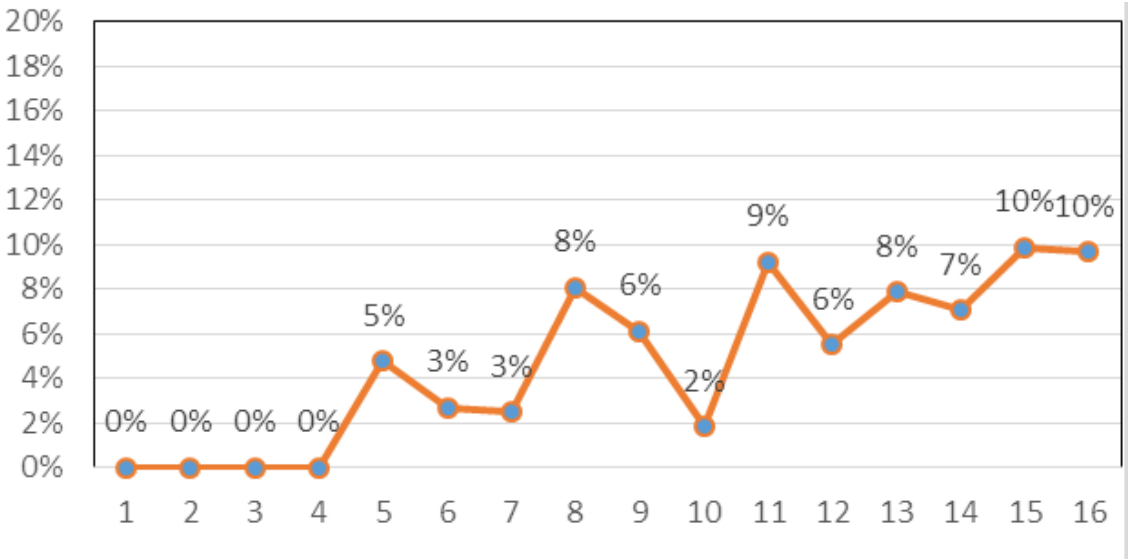


Figure 10. Share of healthcare in process mining publications, per year (2000-2015)

2.3.2 Looking for Existing Literature Reviews

In order to answer research question RQ2_SLR1, we adopted a two-step method. In the first step, we looked for papers that did a literature review on process mining, regardless of any specific domain. Then, in the second step, we selected the papers that are related to the healthcare domain. This method not only answers our question, but also provides us with insight about the endeavours targeting literature reviews on process mining.

Our query was `"process mining" AND (survey OR literature OR review)` and resulted in 37 papers. We excluded papers that:

- Only cited process mining or its derivatives to acknowledge its existence or to mention it in some examples.
- Clearly were related to process mining but were not clearly reviews.
- Were not clearly related to process mining or literature reviews.

Our selection led to eleven papers that reported a literature review of process mining (Table 4). In the following section, we will review these eleven specific papers.

Table 4. Selected literature reviews on process mining, with citations from Google Scholar on October 2, 2021

Paper	*Year*	*Focus*	*Systematic?*	*Source*	*Citations*
Agarwal & Singh [60]	2014	Tools	No	Journal	9
Claes & Poels [32]	2013		No	Conference	74
Tiwari *et al.* [61]	2008	Algorithms and issues	No	Journal	240
Maita *et al.* [62]	2015		No	Journal	35
Breuker & Matzner [63]	2014		Systematised	Conference	10
Yue *et al.* [64]	2011		No	Conference	15
Qing-tian [65]	2007		No	Journal	7
Wang *et al.* [66]	2011		No	Journal	1
Yang & Su [67]	2014	Healthcare	No	Conference	55
Rojas *et al.* [68]	2015		No	Conference	13
Rojas *et al.* [69]	2016		Yes	Journal	438

2.4. Summary of Literature Reviews of Process Mining

After reviewing the selected papers and clustering their main subjects, three main categories emerged: algorithms, tools, and healthcare.

2.4.1 Algorithms

In 2008, Tiwari *et al.* [61] conducted a comprehensive literature review on process mining based on 50+ research papers. They provided an overview of state-of-the-art techniques in order to identify current and future research trends, together with common problems. They also summarised the contribution of some soft computing techniques in process mining. The authors highlighted the application of genetic algorithms [70], general algorithmic approaches [71], Markovian approaches [72], neural networks [73], and cluster analysis [73] in process mining. In 44 of these papers, the share of general algorithmic approaches (other algorithms) in process mining is 62% (see Figure 11). The share of data mining-based approaches and soft computing techniques are only 15% and 9%, respectively. In terms of notations, Petri nets were recognised as the most popular notation, with 29%.

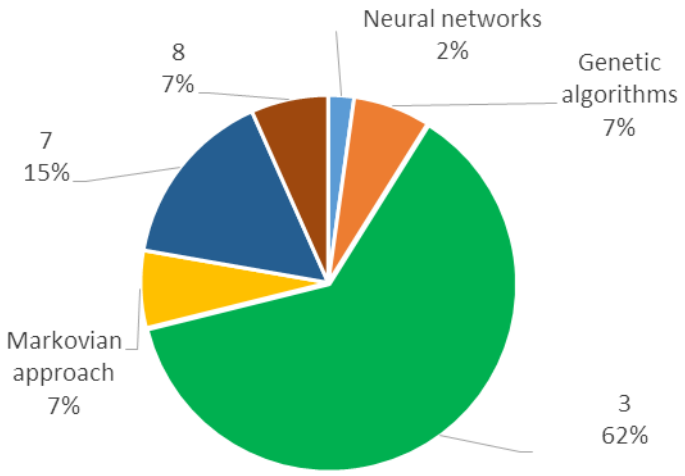


Figure 11. Share of process mining techniques, as reported by Tiwari *et al.* [65]

Tiwari *et al.* [61] also found that some of the papers attempted to address many process-mining problems mentioned in Section 2.1 *Noise* in data is the most common problem reported in the papers (20%) and noise mitigation techniques are widely used to tackle this problem. *Mining loops* and *concurrent processes* are also dealt with by several papers (17% and 15%, respectively). It is worth mentioning that some problems such as *heterogeneous data sources* and *mining perspectives* are not so well investigated, with only one paper each according to Tiwari *et al.* [61]. Although many process mining problems can be addressed using genetic algorithms, the most common problems (noise and mining loops) are still usually tackled with data mining-based approaches.

The concept of organizational routines was developed in management studies and organization sciences to assess collective frequent activity patterns. Breuker and Matzner [63] focused on three kinds of process mining activities: discovery, conformance, and enhancement. Breuker and Matzner reviewed the literature of both fields, i.e., process mining and organizational routines, to clarify their relation. Using Google Scholar and Scopus, they used a systematised approach to search for the papers from the field of organizational routines citing process mining, and they found nine papers. Six papers belonged to organizational routines but cited process mining (including three papers cited in both fields), while the remaining three cited both but did not belong to any of the fields. This review showed that researchers in these two fields employ different research methods and

tools even though they work on similar research questions. This review concluded that using methods from the counter-field could create a great potential for research designs.

Yue *et al.* [64] conducted a review on process mining algorithms. They introduced the work of some scholars and described their algorithms. To this end, the authors have categorised the contributors into Chinese and non-Chinese ones. In this review, 26 papers are studied, and the function of each algorithm is briefly introduced. The algorithms are mostly related to the process discovery topic.

Maita *et al.* [62] focused their recent review on the use of *artificial neural networks* (ANNs) and *support vector machines* (SVMs) in process mining. Eleven papers were selected as primary studies. Their paper concludes that although process mining received significant research interest, less attention has been paid to ANNs and SVMs. This confirms the results of Tiwari *et al.* [61] where only one paper had explored the neural network area in relation to process discovery [72]. The authors believe that this paucity exists mainly because researchers are generally not aware of these two data mining techniques and their potential for solving process mining problems.

Qing-tian [65] studied the detailed definition, research approaches, algorithms, tools, and challenging research problems related with process mining. The author also reported on a subfield of process mining, *distrusted* process *mining*, together with its specific challenges.

Finally, Wang *et al.* [66] considered discovery algorithms of process mining. The algorithms are compared, followed by a discussion of challenging research problems.

2.4.2 Tools

In this context, Claes and Poels [32] conducted a survey about the opinion of people who used the ProM framework and its plugins. Despite the fact that this study is not a systematic literature review on process mining, it is worth considering as it provides a survey on the most prominent tool for process mining (ProM). Inspired by the Process Mining Manifesto, Claes and Poels created a questionnaire about process mining, ProM, and demographics. This survey was answered by 119 people from 26 different countries, with a bias from participants from The Netherlands and Belgium. The authors believe this is mainly caused

by the process mining community having many members from these countries [38]. The authors captured the results of this exploratory survey as follows:

1. Benefits of process mining: objectivity, conformance checking.
2. Drawbacks of process mining: data access (to find and prepare the right data for process mining), too little guidance, hard-to-understand discovered models.
3. Popularity of process mining tools: the ProM framework, Disco.

The first result of this survey is aligned with the result of another survey by Ailenei *et al.* [74] where the same question ("What do you think is the biggest benefit of process mining") was asked from participants.

Aligned with the second result of the survey from Claes and Poels [32], Bose [75] revealed that data quality is the main challenge in process mining research. Regarding the third result of the survey, several process mining techniques used by ProM were reported. The most frequently used plug-ins of ProM are *Fuzzy Miner* [76], *Heuristics Miner* [77], *Social Network Miner* [72], *Dotted Chart Analysis* [78] and the *Alpha algorithm plugin* [79].

In the context of tools, Agarwal and Singh [60] also compared four business process mining tools based on the type of business process mining, process model, process mining problem, algorithm and some additional features. To this end, they used major search engines to find tools capable of performing business process mining. Four tools were selected to be compared: Fluxicon (Disco), ProM, EMiT, and BPM One. Then Agarwal and Singh [60] explained the feature of each tool. The paper, however, does not report a clear comparison of the considered tools.

2.4.3 Healthcare

A clinical pathway clearly determines the goals and key elements of care (e.g., best practices and patient expectations), coordinating roles and sequencing the activities of the multidisciplinary care team. A clinical pathway aims to improve the quality of care, mitigate risks, increase patient satisfaction and increase the efficiency in the use of resources. Several research contributions have identified the lack of a uniformly accepted definition of a clinical pathway [80]. Based on the literature review of De Bleser *et al.* [81], there are 84

different terms that may be considered as synonyms of clinical pathway, including care map, care pathway, critical pathway, integrated care pathway, protocol and guideline.

Yang and Su [67] focused on clinical pathways in a literature review that explains future directions in that area. This paper and the one from Rojas *et al.* [68] are the only two reviews, resulting from our queries, related to process mining in healthcare. In Yang and Su's review, 37 papers published within 2004–2013 were selected and analysed. These papers are categorised in three areas: i) process discovery for clinical pathways design, ii) variants analysis and control, and iii) continuous evaluation and improvement.

In the first area, Yang and Su [67] analysed the difficulties raised through applying *process discovery* techniques in healthcare processes, reported by 19 papers. The difficulties originated form the complex and unstructured nature of healthcare processes, noise, incompleteness, multiple occurrences of activities, and richness of process types and variants, which are some major challenges of mining clinical processes. Yang and Su have concluded that the diversity and complexity of medical processes in clinical pathways are far higher than that of common business processes. The medical environment also has a highly dynamic and uncertain nature, which adds to the complexity. Therefore, mining clinical pathway patterns, using traditional process mining techniques, faces many problems and challenges. Furthermore, the clinical pathways designed using the most recent approaches are non-adaptive and cannot adjust effectively when the external environment changes.

In the second area, 13 papers that mainly use *conformance* techniques and alignment were considered. In these papers, the variants of the clinical pathway are analysed to check the conformance of observed behaviour in the event logs with the model. Considering the techniques adopted in their selected papers, Yang and Su [67] concluded that there are still many technical challenges for the identification and *analysis* of the clinical path variants. They believe that the identification and analysis methods are not always relevant and that they pay little attention to the sources and impacts of the variants, leading to low detection efficiency. The authors recommend that researchers analyse the variants from a systemic perspective, and consider the impacts on the adjustment of the clinical pathways.

In the third area, 11 papers related to *continuous evaluation and improvement* are discussed. They show that many researchers focus on the first two kinds of process mining

(process discovery and conformance), but very limited studies have focused on the continuous evaluation and improvement of actual clinical pathways. According to Yang and Su [67], the evaluation and improvement are regarded as the redesign of the clinical pathways. Nevertheless, the roots that triggered the improvement are always neglected. This neglect will rigorously impact the efficiency and effectiveness of continuous improvement. The authors emphasised that the continuous evaluation and improvement need to be supported by variants analysis. This analysis can find the root causes to ensure the effectiveness and pertinence of improvements.

Finally, the authors state four future research topics to meet the challenges of the medical environment. The further analysis of the variants is the basis for adaptive adjustment and improvement. Variants identification and analysis, customization of clinical pathways, self-learning improvement of clinical pathways, and integrated medical process management are the four key trends identified in the paper.

Rojas *et al.* [68] provided a bibliographic survey about process mining algorithms, tools, challenges, and case studies in healthcare processes. As this review was not done in a systematic fashion, the number of papers in the survey was not clearly mentioned. The publications cited in this paper and referring to at least one algorithm, tool, technique, or case study shows that 30 unique papers are covered by the survey. The paper presents an overview of the characteristics of different approaches including types of processes (three papers), types of data (two papers), process mining tools (eight papers), frequently posed questions (one paper), methods and algorithms (ten papers), used methodologies (three papers), and implementation strategies (seven papers). Moreover, the survey characterises some case studies (nine papers) according to their geographical location and their leading medical field.

Rojas *et al.* [69] extended their previous paper [68] and conducted another bibliographic survey about the application of process mining in healthcare covering 74 papers that report case studies in this context. Rojas *et al.* [69] aim to provide a useful overview of the works undertaken in the field of process mining in healthcare and to help researchers to select process mining algorithms, tools, techniques, methodologies and approaches for their studies. To this end, they tried to identify and characterize the case studies, providing

an overview of the state of the art of the field of process mining in healthcare. In addition, trends and challenges regarding the future of process mining in healthcare are discussed.

Rojas *et al.* [69] have not sensitized or summarised the papers covered by their literature review; they have followed a systematic approach to search, find, include, and exclude papers. In contrast with Rojas *et al.* [68], the methodology and the steps for selecting the papers for review are clearly described. They have looked for a combination of the keywords "process mining", "workflow mining" and "healthcare" using three search engines; PubMed, DBLP, and Google Scholar. In terms of inclusion and exclusion criteria, Rojas *et al.* [69] included the articles that consist of a case study where process mining techniques or algorithms have been applied in the healthcare domain, published in English before February 2016. They excluded the papers that do not present a case study of process mining in healthcare. Moreover, the case studies concerning *Clinical Pathways* have been excluded. After removing duplications and considering the repository of the Health Analytics using Process Mining Team [82] from the Eindhoven University in the Netherlands, Rojas *et al.* [69] finally selected 74 papers for their analysis: 22 papers from the web searches, 16 papers from the repository, and 36 from both places.

Aligned with what we found within our study, Rojas *et al.* [69] reported that there is no comprehensive review of all case studies where process mining has been applied in the healthcare domain.

Rojas *et al.* [69] did not aim to describe the contents of the papers they reviewed and consequently what has happened in the selected papers is not uncovered by their review. The approach characterizations from the two surveys of Rojas *et al.* [68], [69] are as follows:

- **Types of process mining:** In terms of types of process mining (Section 2.1.2), Rojas *et al.* [68] discussed three papers about process discovery and two about conformance checking. Based on Rojas *et al.* [69], 45 papers out of 74 (60% of the case studies) have been related to *process discovery* concerning control flow and discovering the sequence of process activities. *Conformance checking*, which detects actual process deviations from pre-determined model, has been used in 16 papers (21% of the case studies). The performance perspective, that allows improving the models based on analysing the ex-

ecution time, identifying bottlenecks, idle time and synchronization time, has been applied in 10 papers (16% of the case studies). Finally, the organizational perspective, which is based on analysing the collaboration between different resources, has been applied in nine of the 74 case studies (12% of the case studies). Some papers have been considered in more than one of the aforementioned categories.

- **Types of Processes**: Rojas *et al.* [68] observed that processes in the healthcare domain are categorised into two types. The *medical treatment processes* include activities from diagnostics to the execution of patient relief activities. The *organizational processes* focus on the organizational process, the collaborative information of the healthcare professionals and their organizational units. Three papers reviewed by Rojas *et al.* [69] specify this classification of processes. However, another classification presented in one paper divided operational healthcare processes into two types: *non-elective* care, including medical emergencies; and *elective* care, which includes scheduled standard, routine and non-routine procedures.
- **Types of Data:** The types of data determine the processes capable of going through analysis. Also, the event logs are built using the content of these data. Rojas *et al.* [69] found only two case studies representing classifications of data. The first one categorized the data into *vital signs, events,* and *personal data of patients*, whereas the second classification is established regarding the data source and its level of abstraction, accuracy, granularity, directness, and correctness. This latter classification uses four data sources: *administrative systems*, *clinical support systems*, *healthcare logistic systems*, and *medical devices*. A clear understanding of the available data types supports researchers in building event logs and applying process mining in a correct manner [69].
- **Frequently Asked Questions:** Mans and van der Aalst [83] defined four typical questions to be answered by medical process specialists: i) What are the most followed paths and what exceptional paths are followed? ii) Are there any differences in care paths followed by different patient groups? iii) Do we comply with internal and external guidelines? iv) Where are the bottlenecks in the process? Rojas *et al.* [68] proposed a fifth question in order to include the organizational collaboration process among specialists: What are the roles and social relationships between the medical staff? Two

studies reviewed by Rojas *et al.* [69] presented a classification of frequently asked questions (3% of case studies).

- **Techniques or Algorithms**: Rojas *et al.* [68] presented seven algorithms implemented to execute the desired analysis in process mining in healthcare. Most were discussed in the previous reviews: Trace Clustering (two papers), Fuzzy Miner (two papers), Alpha Miner (one paper), Genetic Miner (two papers), Heuristic Miner (three papers) and Conformance Checker (two papers). However, one algorithm was original: Performance Sequence Analyzer (two papers). Rojas *et al.* [69] extracted 33 techniques or algorithms used in the case studies they reviewed and showed the number of cases that used each algorithm. The most commonly used techniques are Heuristics Miner (19 papers, 26%), Fuzzy Miner (15 papers, 20%), and Trace Clustering [84] (8 paper, 11%). This result is aligned with the result of Claes and Poels [32] described in Section 2.4.22.4. As Rojas *et al.* [69] showed, 19 algorithms that were extracted from 74 case studies are used only in one paper, so there is room for additional studies.
- **Implementation Strategies:** One original contribution of Rojas *et al.* [68] is that they classified the strategies used to implement process mining in three classes based on their *level of automation*. The first one is the basic direct strategy, which utilises the process mining tools directly with an event log extracted directly from a data source (four papers). The second strategy is semi-automated, where a specific solution is needed to connect to one or several data sources and extract the data to build the event log (one paper). The third strategy is more automated. Here the steps are done in a specific suite, including connection to the data sources, extraction of the data, building of the event log and implementation of the process mining techniques. Here the person using the suites does not need knowledge of process mining tools in detail, but the suites are developed exclusively for an environment and its data sources. Medtrix Process Mining Studio [85] and the Emotiva Tool [86] are two examples of such suites. In terms of the implementation strategies, Rojas *et al.* [69] show that the first strategy (the direct one) is adhered to in 17 case studies (23%), whereas the second strategy (the semi-automated one) is applied in only one case study. Finally, the third strategy (automated) is applied in 7 case studies (9%).

- **Process Mining Tools:** In Section 2.5, we described the main tools available for process mining in general. Rojas *et al.* [68] focused on the healthcare domain and considered seven papers to find the frequency of some main tools. They showed that ProM [28] has been used in a majority of case studies in healthcare. They, also, considered the other popular process mining tool Disco [33], and showed that it is not mentioned as much as expected in the case studies that they reviewed. Aligned with these results, Rojas *et al.* [69] concluded that ProM is the most common process mining tool used in healthcare, used in 31 of their case studies (42%), followed by Disco (8 case studies, 11%).
- **Types of Case Studies:** Rojas *et al.* [68] defined three types based on the process mining tool they have used. The first type is the basic case study, in which there is no new implementation done and the objective is to provide knowledge of the healthcare process. A majority of the case studies have been related to of this type. In the second type, a new technique or algorithm has been developed as a complement to current tools. The third type is a mixture of tools and techniques used in cooperation with some techniques from other fields such as statistical analysis and data mining.
- **Geographical Analysis of the Case Studies:** Aligned with our results from Section 2.3.1, Rojas *et al.* [68] show that use of process mining, as an important tool for analysing medical processes and generate improvements opportunities, has a growing trend in the recent years. Based on a quantitative count and *geographical* classification on the case studies available, the highest share of case studies is in Europe. There exist only a few in North America, Asia, and Australia, and none in Africa or South America. Within the 74 case studies included by Rojas *et al.* [69], around 73% are in Europe, with a few examples in Australia, Asia, and North America. Specifically, in Europe, The Netherlands is the country with the highest number of case studies, followed by Germany and Belgium.
- **Medical Fields:** Based on Rojas *et al.* [68] in terms of the *fields* covered by the case studies, most have been done in oncology and surgery, and a few others in caregiving, cardiology, diabetes, dentistry, medication, intensive care, and radiotherapy. One year later, Rojas *et al.* [69] identified 22 different fields where data were gathered. Oncology

(9 case studies) and Surgery (8 case studies) are again the medical fields with the highest numbers of case studies. Analysing the distribution of case studies within 22 medical fields shows the multidisciplinary character of process mining in healthcare, together with its potential application to all medical fields [69].

- **Analysis Strategies:** Rojas *et al.* [69] established the use of three analysis strategies from the literature review and the case studies identified. These strategies are based on the way in which the task of applying process mining techniques and algorithms is undertaken. In the first strategy, namely *basic strategy*, the process mining techniques and algorithms prevalent in the available tools are applied. This is the easiest strategy to perform without using any new technique or algorithm, or techniques from other areas (15 papers, 19% of the cases reviewed). In the second strategy, a new process mining technique or algorithm with an objective particular to each specific case study is implemented. The main objective with the new implementation is to discover novel ways to deal with processes that are flexible, unstructured and complex, and also to handle large, multidimensional datasets (6 case studies, 8%). In the third and last strategy, in addition to current techniques and algorithms in current tools, researcher incorporates analysis of other areas, such as statistical analysis (10 papers, 13%), data mining (4 papers, 5%), ontologies (2 papers, 3%), or simulation models (1 paper, 1%). It is noteworthy that the resource consumption, the complexity, and the required expertise in the second strategy are higher than for the others. Yet, the third strategy is also challenging because forming a team of analysts using the techniques from several areas is required [69].
- **Challenges:** The limitations identified by Rojas *et al.* [68] relate to data (seven challenges) and to the involvement of expert medical knowledge (two challenges). Aligned with the results of Tiwari *et al.* [61] in Section 2.4, the limitations related to *data* are the main challenges of process mining even in the healthcare domain: identifying and accessing data sources; including the physical information and conditions of the patients; data integration from different sources; data quality; granularity and pre-processing of the data; using real event logs and data; and building a correct and complete event log. The second category of challenges includes satisfying medical protocols and guidelines, and including medical knowledge. According to Rojas *et al.* [69], one of

the weaknesses of current process mining tools is the absence of a good visualization of the process models, especially for complex and less-structured processes such as those in the healthcare domain. A great amount of reliance on experts for applying process mining is another challenge. The tools or solutions that are straightforward to apply, without the need for deep knowledge of the tools and techniques, are yet to be developed.

The results of these three healthcare-oriented reviews are hence in line with the results from the general process mining reviews, with additional challenges requiring further attention.

2.5. Literature Review of Goal-oriented Process Mining

This section aims to review the papers related to goal-oriented process mining. The studies focusing on the combination of goal modelling and the capabilities of process mining techniques are discussed. The methodology of this SLR, including the research questions, the identification of search engines, the selection criteria, and the inclusion/exclusion criteria are provided. Then, among the selected papers, those that are related to the topic of this research are categorized and their contributions are summarized.

Inspired by the work of Kitchenham *et al.* [59], this systematic literature review aims to answer the following research questions:

- **RQ1_SLR2.** What is the share of goal-oriented approaches in process mining studies? This question aims to determine whether goal-oriented process mining is a mature research field or whether there is currently a gap that exists.
- **RQ2_SLR2.** How are the studies conducted on the joint goal modelling and process mining techniques categorized, and what are the main contributions of the papers in these categories?

2.5.1 Identification of Search Engines

The literature review from the previous section (Table 2) showed that nearly 96% of the process mining papers were covered by Scopus and Google Scholar (with very few additional unique papers from the other six engines used, mainly by IEEE Xplore and ASC).

However, to be on the safe side, in addition to Scopus and Google Scholar, IEEE Xplore and PubMed are also used in this second literature review. As this review targets the papers positioned at the intersection of the process mining and goal modelling areas, the desired papers are necessary positioned in the process mining area.

2.5.2 Keywords and Queries

As we are looking for papers focusing on process mining and goal modelling, we use two sets of keywords, i.e., one for each area. The following queries were defined in all four search engines looking for papers that include at least one of the keywords from each set:

```
(goal[Keywords] OR "goal-oriented" OR "Goal modelling" OR "Goal
modelling" OR "Goal model" OR "Goal mining" OR "Goal monitoring" OR
"requirements engineering" OR "user requirements notation" OR KAOS
OR istar OR "i-star" OR "i star" OR "NFR Framework" OR Tropos OR GRL
OR intention OR intentional OR "performance indicator" OR KPI)

                              AND

("Process mining" OR "Process discovery" OR
   "conformance checking" OR "event log")
```

Note that this query was constructed iteratively, i.e., we started with some essential keywords such as goal modelling, and then the returned papers informed us of other relevant and related keywords such as intention and requirements engineering.

Initially, the query within the full text of the papers was run. This query returned thousands of papers, but most of them were irrelevant. In order to end up with manageable, meaningful, and relevant results, the search was limited to metadata (title, abstract and keywords). As searching within abstracts and keywords is not supported by Google Scholar, queries on that engine were limited to the title only (patents and citations were also excluded). Also, given that the word "*goal*" is a popular word used in almost all paper abstracts (e.g., "*the goal of this paper is...*"), the queries were limited to look for this specific word only within the keywords (the other search terms were searched for title, abstract, and keywords). This constraint reduced the number of results dramatically and enabled the authors to analyze the related papers while avoiding noise from less relevant papers. To conserve the ability of finding as many papers as possible, the review was not consciously limited to a particular time period.

The results of the query are shown in the first row of Table 5. Finally, after removing duplicates among the results of the four search engines, 92 papers remained.

Table 5. Number of papers returned by the search engines

	Search within	*Google Scholar*	*Scopus*	*IEEE Explore*	*PubMed*
1	Limited to title/abstract/keywords	11	81	10	7
2	Duplicates removed, across all engines		*92*		

2.5.3 Frequency Analysis of Publications

In order to find the position of *goal-oriented modelling* in the process mining research area (and vice versa), two more queries were run with Scopus looking for the aforementioned sets of keywords separately. This time, the keywords were `{"goal-oriented" OR "goal model" OR "intention mining" OR "user requirements notation" OR KAOS OR istar OR "i-star" OR "i star" OR "NFR Framework" OR Tropos OR GRL OR "Goal mining" OR "Goal monitoring" OR "intentional modelling" OR "Goal modelling" OR "Goal modelling"}`, (this time only) within the indexed keywords of journal and conference papers. `KPI` and `"performance indicator"` were omitted because these keywords independently do not refer to the goal modelling field. Also, the same engine was queried run with the keywords `{"Process mining" OR "Process discovery" OR "conformance checking" OR "event log"}`. The numbers of papers returned through each query were 3048 and 2203, respectively. These high numbers clearly show that the papers at the intersection of these areas of research (92) are quite sparse.

In addition, the authors considered the frequencies of all keywords that have been indexed by the 2203 papers related to process mining returned by Scopus. It is not surprising that the keywords "*process mining*" and "*data mining*" are the top ones, by far. However, the word "goal" did not show up within the top 160 keywords that were ranked based on frequencies.

Table 6. Selected papers related to goal-oriented process mining, with citation numbers from Google Scholar on March 28, 2018

Category	*Paper*	*Code*	*Makes reference to*	*Year*	*Source*	*Cited*	*Goal model representa-tion*	*Proposed goal mining algorithm?*	*Case study/ evaluation?*
Goal Modelling & Requirements Elicitation	[87]	P1	-	2012	Conf: ICITCS	0	None	No	No
	[88]	P2	-	2012	Conf: ICCSA	0	None	Yes	Yes
	[89]	P3	-	2014	Journ: TMIS	3	AND/OR graph	Yes	Yes
	[90]	P4	-	2015	Conf: SEKE	0	AND/OR graph	Yes	Yes
	[91]	P5	P12	2015	Conf: BPM	1	KAOS	No	Yes
	[92]	P6	-	2016	Journ: SoSyM	9	None	No	Yes
	[93]	P7	-	2017	Conf: BPM	1	None	No	No
	[94]	P8	-	2017	Conf: ER	0	AND/OR graph	Yes	Yes
	[95]	P9	-	2017	Conf: RE	0	None	No	Case study
	[96]	P10	-	2017	Journ: J Healthc Eng	0	None	Yes	Yes
Intention Mining	[97]	P11	P12	2013	Conf: RCIS	5	Mentioned and compared some of them		Yes
	[98]	P12	P13, P11	2013	Conf: RCIS	21	MAP	Yes	Yes
	[99]	P13	P11, P12	2013	Conf: EMMSAD	18	Mentioned and compared some of them		Yes
	[100]	P14	P11, P12, P13	2014	Conf: RCIS	2	MAP	Yes	Yes
	[101]	P15	P11, P12, P13	2014	Journ: IJISMD	2	MAP	Yes	Yes
	[102]	P16	P11, P12, P13	2014	Conf: MSR	15	MAP	Yes	Yes
	[103]	P17	P12, P16	2014	Conf: BPMDS	9	MAP	Compare	Yes
	[104]	P18	P11, P12, P13, P14, P16, P17	2014	Thesis: U. Paris-Est	3	MAP	Yes	Yes
	[105]	P19	P12	2014	Conf: CAiSE	5	MAP	Yes	Yes
	[106]	P20	P14, P15, P16, P17, P18	2015	Journ: IJISMD	2	MAP	Yes	Yes
KPI	[107]	P21	-	2013	Conf: CBI	7	Performance analysis		Yes
	[108]	P22	P21	2014	Conf: CBI	2	Performance analysis		Yes
	[109]	P23	-	2017	Conf: OTM	0	Model redesigning with KPIs		Yes
	[110]	P24	-	2017	Journ: DSS	0	Performance analysis		Yes

2.5.4 Selection Criteria and Related Papers

In order to exclude the papers that are irrelevant to our two research questions, exclusion criteria were defined. These criteria helped exclude papers that:

- only cited process mining and goal modelling or their derivatives to acknowledge their existence or to mention them in passing in some examples; or
- were related to only one or none of the two areas considered; or
- were not written in English.

The use of the above criteria resulted in 24 papers (including one thesis) in which a research activity related to the topic of this review was reported (see Table 6). A quality assessment confirmed that the papers (especially those returned by Google Scholar) had sufficient scientific content and were not coming from predatory journals or conferences. In the following section, these specific papers will be reviewed.

2.6. Analysis of Selected Papers of the Second SLR

After considering the selected papers and clustering their main subjects, the papers were categorized into three main areas: Goal Modelling and Requirements Elicitation, Intention Mining, and Key Performance Indicators (KPIs). In this section, the selected papers are analyzed, and their contributions are summarized.

2.6.1 Goal Modelling and Requirements Elicitation

Business processes are designed to achieve certain goals, namely *process goals*. Based on these process goals, individual agents are assigned to perform the related tasks composing the process. However, the agents generally belong to different sections/units and have different interests, and their behaviour is not usually observable. In the real world, agents have intentions reflecting their own interests, namely *agent goals*. In order to satisfy these goals, the agents may not always behave according to a designed process. The inconsistencies or, in some cases, conflicts between process goals and agent goals often result in a reduced efficiency or effectiveness of the executed process [89]. However, traditional business process management ignores the feature of self-interest in the business process and adopts an

inter-organization workflow perspective [111]. This phenomenon is common in the context of agent-oriented business processes, which have been the focus of the literature in some research domains such as healthcare [112], supply chain management [113], financial fraud management [114], and others. Process mining could have a valuable contribution in addressing this common problem. However, in the literature, there is a lack of studies on how to discover agent goals in business processes. To fill this gap and to address that problem, Yan *et al.* [89] proposed an agent-oriented goal mining approach for modelling, discovering, and analyzing agent goals in executed business processes using event logs. This is clearly positioned in the context of process discovery as a type of process mining activity.

Yan *et al.* [89] used the classical goal reasoning framework of *belief-desire-intention* (BDI) logic [115] to analyze an agent's choices using event logs with domain data from the business process. They have explained some supporting reasons to show that the BDI model has several advanced features that meet the requirements of an agent's goal mining, compared to other computational decision models. Yan *et al.* [89] have used the framework shown in Figure 12 within a sample process used to invite external reviewers to evaluate loan requests.

Three activities are executed in the process. First, an agent assigns a reviewer for each loan request. Then, a reviewer performs the activity of evaluating the request, and finally the request is either assigned to a new reviewer or sent to the payment agent. The authors have elaborated this example, which has been supported by some event logs used to mine the agent goals (number 3 in Figure 12).

In their proposed goal mining algorithm, the goals are expressed as decision tree rules. In order to discover an agent's goal, Yan *et al.* [89] adopt an approach to discover the agent's desire-accessible worlds based on the agent's previous activities. They assume that if an agent has chosen an activity, she knows the effects of the activity and believes that she achieves her goal by performing this activity. Thus, one can learn the agent's goal by classifying the agent's activities in different situations. In other words, the agent's desire-accessible worlds can be identified by machine learning methods. Yan *et al.* [89] adopted the decision tree algorithm (C4.5) to construct the desire-accessible worlds, i.e., the agent's goal, within their example. As inputs, their proposed algorithm needs event logs

and the agent-oriented Petri net process model discovered by process mining algorithms. The outputs of the algorithm are the agents' goals at each state.

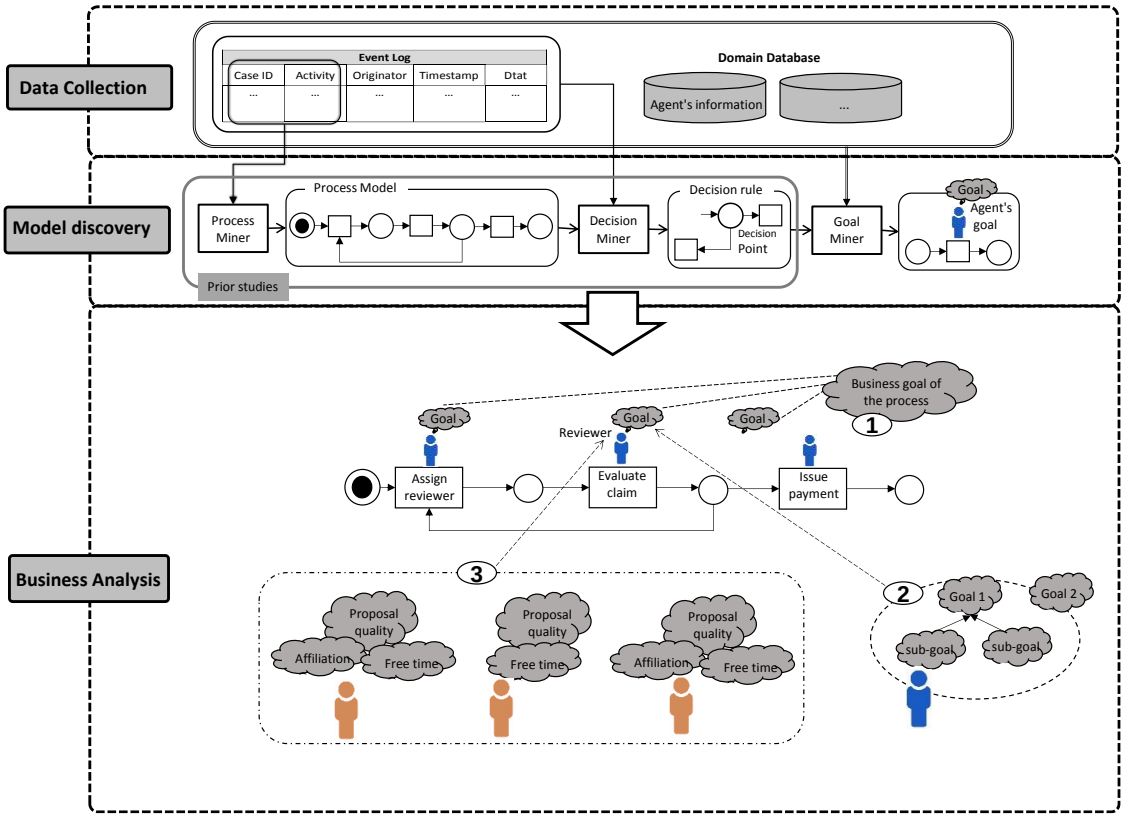


Figure 12. Yan and Hu's framework proposing an agent goal mining approach. Adapted from [93].

Yan *et al.* [89] have evaluated their framework by simulation techniques and compared their goal mining-based approach with the best-performer-based approach, using t-test for testing the improvement hypothesis. Based on their case study, in terms of improving the effectiveness (=1 – Number of requests wrongly graded / Number of requests), their goal-mining-based approach has not significantly improved effectiveness, comparing with the best-performer-based approach. However, their approach has significantly improved efficiency (=Number of requests / Evaluation times).

Along with the goal-oriented approach of Yan *et al.* [89], Armentano and Amandi [88] focused on the automatic recognition of the goal that motivates an employee (agent)

to perform a particular sequence of tasks. This is crucial to determine what tasks are expected to be executed next in order to achieve the goal within the dynamics of the organization.

Traditional process discovery techniques [72] do not focus on detecting the goals that motivate actors to do their tasks. Nevertheless, the line of work of Armentano and Amandi [88] corresponds to the area of *goal recognition*. In this direction, they considered the prediction of the subject's goal as an inherently uncertain task. Therefore, they looked for a knowledge representation able to capture and model this kind of uncertainty.

The Markovian approach, proposed by Cook and Wolf [116], is one of the traditional approaches for process discovery. The approach presented by Armentano and Amandi [88] differs from the Markovian approach in that the latter one used Markov chains of first and second orders, while the former one used variable-order Markov models.

In this proposed approach, the model is constructed from the observation of the tasks that an employee executes and from the goal that motivates the execution of those tasks. The model predicts the most probable agent's "*class*" (*goal*) after each performed activity. A learning algorithm is used to make a model of the tasks necessary to achieve different goals. This knowledge could be preserved in the organization, contributing to the organizational memory. Finally, this can contribute to detecting the goal of an employee at any time and can help identifying deviations from expected behaviours [88].

Normally, any industry-scale business process takes different variants that ideally contribute to achieving the goals of an organizational goal model. Consequently, the adequate management of process models with many variations mandates considering each process variant as an independent model entity. Ponnalagu *et al.* [91] worked on the admission of a deviating process instance as a valid variant of the intended process that achieves the same goals as the intended process. They focused on concept and goal conformance drift that rises with evolutionary changes in process executions. This could be admitted as a crucial aspect especially in knowledge-intensive processes, where significant drifts from valid (goal aligned) workflows could happen due to manual errors and environmental constraints.

There exist studies that worked on the categorization of processes [117], discovering a family of process variants from a collection of event logs [118], and on the management of large collections of process variants of a single process model [119]. In their approach, Ponnalagu *et al.* [91] validate and categorize the discovered family of variants based on a goal model. Their approach can leverage the works that focus on enriching process designs with goal-driven configurations [120]. Categorizing process execution deviations in a goal-based fashion is necessary to decide if a deviation represents a valid variant. Ponnalagu *et al.* [91] proposed an approach to help with the decision of whether process instances in execution logs are valid variants, using a goal-based notion of validity. Their approach also supports the analysis of the impact of contextual factors in the execution of specific goal-aligned process variants. They used the KAOS methodology for goal decomposition. Moreover, they examined their approach with an Eclipse-based plugin and, to validate their approach, they considered a process log of 25,000 events obtained from the history of a help desk division in an IT organization. They demonstrated that with semantic annotations of process models, goal models in terms of the end effects are accurately applied for a given domain. Also, in terms of being the right candidate for dealing with a larger data set composed of processes, their approach is general enough, i.e., it is not restricted to goal models or process models of any scale or domain.

As explained previously, *conformance checking* is acknowledged as one of the main categories of process mining activities, where the real executed process is compared to the process model in order to find and analyze deviations. There are many conformance checking approaches mainly focusing on aspects of business processes, but this is not adequate for analyzing whether the actual business processes are able to satisfy organizational goals.

According to Horita *et al.* [90], goal-oriented requirements analysis methods received notable research interest in requirements engineering. This approach is useful for reflecting organizational requirements into business process models, but actual business processes usually deviate from defined process models. Hence, it is important to analyze the data recorded as event logs along with using the model's information. Horita *et al.* [90] proposed a goal-oriented conformance checking approach that not only can detect deviations between logs and models, but can also analyze the effects of a deviation.

As shown in Figure 13, the method proposed by Horita *et al.* [90] consists of two main phases. The first phase, called Trace & Goal Processing, checks whether event logs satisfy the goals of a goal model represented as logical formula, and constructs cross-tabulation tables for the second phase. All traces are divided into goal satisfied traces or not satisfied traces. The numbers of traces when a goal is or is not achieved compose these cross-tabulation tables. The second phase, called Statistical Analysis, considers the relation between the goals and analyzes the positive effects or negative effects of a violated goal on the other goals in the goal model. This phase uses statistical hypothesis testing on cross-tabulation tables constructed in the first phase as an input, in particular Chi-squared test and Fisher's exact test. The authors have applied their approach on an event log of a phone repair process.

In [93, 121], Bernard and Andritsos focused on *Customer Journey Mapping* (CJM) as an emerging research area. In particular, CJM tracks and describes customers' responses

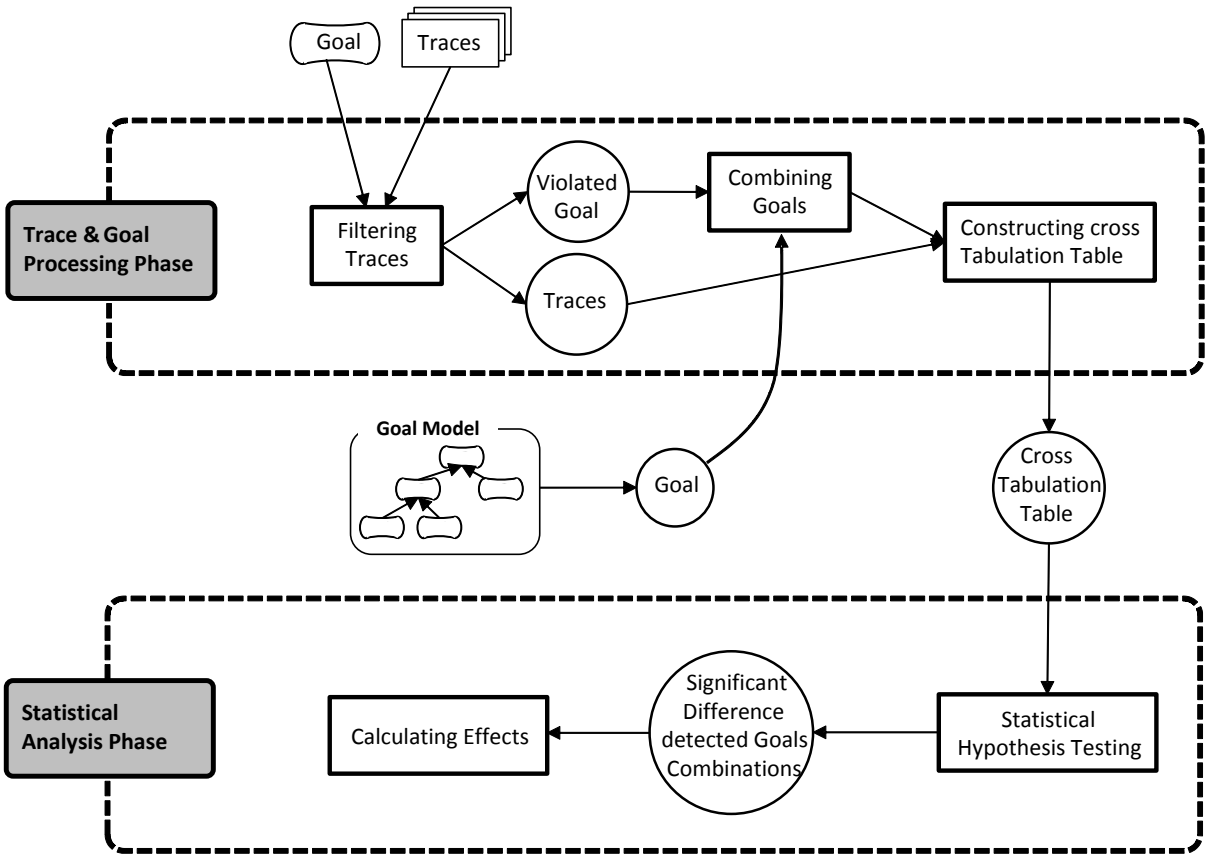


Figure 13. Overview of the goal-oriented conformance checking method proposed by Horita *et al.*, adapted from [94]

and experiences when they are using a service and represents them, as *customer journeys*, on a map [93]. The challenge with CJM is that it is not clear how to present thousands of customer journeys. Inspired by process discovery techniques, borrowed from process mining, Bernard and Andritsos developed the CJM-explorer (CJM-ex) tool. This tool is a web interface that enables interactive navigation through several journeys stored in event logs. To this end, the application uses hierarchical clustering, statistical indexes, and *user-defined goals*. CJM-ex aims to show how the event logs can be displayed on CJMs, and to let analysts navigate through these journeys in a goal-oriented fashion.

In the proposed method, a hierarchical clustering algorithm is first used to segment the original customers' traces into groups of similar journeys. After forming the clusters, CJM-ex becomes able to leverage the contextual information (e.g., emotions or characteristics) associated with a typical customer journey. The clustering algorithm generates a tree (or dendrogram) where the journeys seen in event logs are at the leaf level and get merged at higher levels to form "*representative*" journeys.

From a goal-oriented perspective, this method does not deal with customers' goals or intentions. However, it lets the users of the tool (e.g., analysts) define their own exploration goals. For example, an analyst might be interested in journeys that started by a specific activity (e.g., "calling the customer service") experienced by customers with specific characteristics (e.g., people younger than 16). Based on the goal setting, important branches of the clustering tree will be highlighted in "hot" areas of the tree. This feature makes CJM-ex the first goal-oriented tool that enables analysts to set a-priori goals to guide journey exploration.

As explained in Section 2.2, *goal models* play an important role in requirements engineering by representing statements of stakeholders' intent in a hierarchically form. Here, the goals in higher levels of the hierarchy (parent goals) are related to goals in lower levels (sub-goals) via AND or OR refinement links. In that context, Santiputri *et al.* [94] aimed to address the question: *can enterprise goal models be mined from readily available enterprise data?* Note that the method they proposed does not consider mining *goals* directly from event logs. Rather, it aims to leverage operational data that manifest pre-deployed goal refinements. In summary, this method involves mining event logs to leverage

so-called *temporal correlation patterns* between goals and sub-goals, mining goal refinement patterns from multi-layered event logs, and composing goal refinement patterns to obtain goal models (or goal trees). Temporal correlation patterns indicate that the achievement of a goal relates to the achievement of its sub-goals. Such a pattern requires that in a sequence of events, the events denoting achievement of a parent goal occur immediately (or soon) after the events denoting the achievement of its sub-goals. For example, the event that denotes occurrence of "making a cup of tea" (as a parent goal) occurs immediately after the events that denote "putting a teabag in a cup" and "pouring of hot water into the cup" (as its sub-goals).

In this approach, it is critical to assume that the input event logs are partitioned into different levels of abstraction. This key assumption suggests that the events implying achievement of parent goals appear in the log of more abstract events (or higher-level), while the events implying achievement of sub-goals appear in the log of more refined events (or lower-level).

The next step is to compose goal refinement patterns to obtain goal models. Here the main challenge is the difficulty in relating semantically similar, but syntactically highly dissimilar requirements of goals and sub-goals. For example, a sub-goal may be represented as: *log labor hours for billing*. Pretty distinctly, we might mine a goal refinement pattern for a parent goal represented textually as: *track technician time for charging the customer*. Human intuition advises that these two goals are semantically similar, and any existing know-how for the latter is also useful for the former. To cope with this challenge, Santiputri *et al.* [94] use a state-of-the-art mechanism for measuring syntactic and semantic similarities of words and phrases, called *word2vec* [122]. This mechanism consists of a two-layer neural network that returns a real-value measure of semantic similarity of two given words (the higher the value, the more similar). Hence, using an appropriate threshold defined by domain experts, one can relate a phrase describing a sub-goal in one goal refinement pattern to a phrase describing a parent goal in another goal refinement pattern. This approach does not guarantee the accuracy of all mined goal models and hence analysts' oversight and editing are still required. This method helps modellers improve their efficiency as they can use the "first draft" models or model fragments (instead of a "blank sheet") to design usable models.

This approach was evaluated using both real-life and synthetic datasets. Overall, the results of the empirical evaluation suggested that the combination of two proposed techniques (sequential pattern mining for leveraging temporal correlations patterns and word2vec for evaluating goal/sub-goal similarity) is a promising basis for goal model mining.

In [95], Dabrowski *et al.* took into consideration a common problem of software designers during the elicitation of software requirements, and propose a solution based on a goal-oriented process mining approach. Software designers have to make assumptions (often invalid) about the users' processes and consequently about the requirements supporting such processes. Elicitation and validation of such assumptions manually through interviews, observations, etc. is time consuming and expensive. Furthermore, designers may fail to find the users' real processes. To address such a problem, Dabrowski *et al.* [95] proposed a new approach that leverages the synergy between *Requirements Engineering*, *Crowdsourcing*, and *Process Mining*. Such a combination is employed to identify and validate software process requirements exploiting the crowd's perception [123], [124]. This approach takes advantage of the capabilities of process mining to discover the underlying processes of crowds from event logs. A crowd brings an opportunity to involve large and diverse groups of users to interact with a given system and conduct their intended processes leading to achievement of their goals. Then, process mining techniques can discover such underlying processes from the logs stored in databases. The discovered processes become implicit feedback used for revising the existing system's functionalities and for ensuring that a software system operates as expected. In addition, missing functionalities and quality issues may be revealed by the analysis of user-system interactions.

The novelty of this approach pertains to its ability to extend requirements engineering techniques by revealing new requirements using process mining techniques along with crowdsourcing. This is different from other crowdsourcing techniques in the literature that are mainly focused on extracting explicit user feedback [123], [125].

The work of Dabrowski *et al.* [95] aims to answer the question "*how to discover requirements through goal-driven process mining?*". They explained their ideas through a simple example (a personal energy management system) as a case study and illustrated the motivation behind the approach. First, end-users execute their daily processes using the

application for a given period of time. Then, the corresponding usage log is input to process mining algorithms to extract the underlying process models. The purpose of this process mining step is to reveal typical patterns of user behaviour, process bottlenecks, and misalignments or variants between reality and intended nominal processes. This typically benefits a software engineer that has problems detecting discrepancies between real processes and designed ones [126]. Such a benefit is possible by conformance checking and denoting the points in the process not aligned with the high-level user goals [95]. This step results in a list of possible new requirements to address misalignments between designed and actual processes. Further analysis here could lead to the identification or/and confirmation of new stakeholder goals and underlying requirements. The resulting artifact then forms an input for the *planner* who is responsible of identifying tasks and objectives to be delegated to the *crowd*. In the proposed approach, "crowd" denotes a large and diverse group of users delegated to perform some processes using the given application to accomplish predefined objectives [123]. The planner then orchestrates objectives and tasks to be carried out by the crowd.

The crowd interactions with the software are similarly stored as logs and subsequently analyzed. If a misalignment is still present, then this may lead to the implementation of newly found requirements in the application. The synergy between crowd and process mining techniques can indeed become an important source for new requirements.

The work of Dabrowski *et al.* [95], which exploits users' real behaviours for detecting misalignments, can effectively contribute to requirements engineering and, also, to process improvement. This contribution is promising not only in software systems but also in organizations striving to improve compliance and design better processes.

In this regard, Outmazgin and Soffer [92] proposed a process mining-based analysis to study *intentional incompliances*, where employees intentionally deviate from the required procedures even if they are aware of them. Understanding intentional incompliances, or *work-arounds*, can improve an organization' performance by helping redefine requirements and redesign processes and support systems.

There exist numerous research activities about compliance conducted separately within the communities of requirements engineering [127], [128], [129] and process mining [130], [131], [132]. Existing process mining techniques for compliance checking do

not distinguish intentional incompliance and do not address their reasons and sources. In contrast, Outmazgin and Soffer [92] defined a list of generic work-around types found in real practices. They discussed whether and how these work-arounds can be detected by process mining, and they analyzed their sources.

A work-around denotes the employees' perception of prescribed procedures as an obstacle for some of their desired goals, when intentionally not following these procedures. Work-arounds are usually recognized as a negative phenomenon, assuming that the prescribed process is designed and optimized to reach the desired business goals. However, since work-arounds are intentional behaviours of employees, they are performed for certain reasons. Poelmans [133], [134] reminds that users work around a system for the purpose of saving time and/or efforts or avoiding the system's limitations. Outmazgin and Soffer in [135] show that work-arounds may be triggered when the predefined business processes are not able to accommodate atypical situations that may arise. Moreover, work arounds can be motivated when the designed process or its support system cannot satisfy all the stakeholder expectations. Other cases might occur when employees decide to pursue their own personal goals rather than to follow the process designed based on the organization's goals.

The qualitative study of Outmazgin and Soffer [92] that explores work-arounds in several organizations led to six generic types: A) bypass of process parts, B) selection of an entity instance that fits a preferable path, C) post-factum information changes, D) incompliance to role definition, E) fictitious entity instances, and F) separation of the actual process from the reported one. Then, in [92], they investigate whether and how the work-arounds can systematically be revealed using event logs. To this end, they have used Disco [33] and explored the logs of five processes belonging to three organizations over two years. Outmazgin and Soffer [92] further characterized the log patterns that can be used to detect work-around of types A, C, D, and F. Also, they explained why such pattern cannot be defined for the other two types (B and E). For example, an experienced employee splits purchase requisitions and place several small requisitions, instead of a high-priced one, to avoid long approval paths.

Next, Outmazgin and Soffer [92] used process mining to conduct a quantitative analysis of situational reasons that can be associated with work-arounds. Using the data of

real processes, they have statistically tested the hypothesis of a correlation between different types of work-around and *activity durations*, *number of participants*, and *work handover*. For instance, they found that work-around of type A (bypassing some activities), as the most frequent type, is positively correlated with activity duration and with the actual number of activity participants.

Aligned with these results, a case study in Belgium [134], also suggested that the sources of work-arounds can be mitigated through the following strategies: deliberate efforts to decrease specific activity times, improving control and designing appropriate work handovers, enlightening access permissions to prevent unauthorized employees doing activities, and monitoring work-arounds and performing disciplinary actions. These practices are expected to lead to performance and compliance improvement.

One of the principal domains where process mining capabilities are leveraged is healthcare [6]. In particular, clinical pathway (CP) mining from historical data has received increasing attention. Some approaches focus on using *topic modelling* methods to discover clinical patterns instead of process models. The topics are semantic compositions of clinical activities often discovered using Latent Dirichlet Allocation (LDA), a statistical model developed for text mining and machine learning first presented by Blei *et al.* [136]. Xu *et al.* [137] used topic modelling together with process mining to generate a concise and interpretable topic-based process model. The key principle in their work is to discover the latent topics from clinical data using LDA and, then, to extract the order relations between these topics using process mining techniques. In this approach, treatment activities are mapped to words and the latent clinical topics are mapped to latent topics of texts.

Xu *et al.* [96] further aimed to consider those latent clinical topics as *clinical goals*. They assumed that the clinical activities taking place in a specific time interval (usually a hospitalization day) are prescribed for multiple clinical goals, and each clinical goal corresponds to a set of clinical activities. Finally, a process discovery framework is applied on the goal-based sequences (instead of the detailed clinical activities) to get a comprehensive process model [96]. In order to evaluate the proposed approach, they used two real-world datasets. The results show that the topics revealed by their method provide higher coherence, informativeness, and coverage than the raw LDA. These extracted high-quality topics

are proper for representation of the clinical goals. Also, the method is effective in generating a comprehensive topic-based clinical pathway model.

Baek *et al.* [87] focused on the adaptability of the processes discovered by process mining techniques in terms of being dynamically reconfigured for new requests made on business processes. They proposed an analytical method using a heuristic algorithm. This algorithm is based on the goals to create an adaptive process mining model. The model is able to provide a continuous service demand scenario that is created dynamically. Baek *et al.* proposed some measurements to calculate the correlation among the activities or the importance of each activity within a given process and the similarity of some processes. They took advantage of concepts analogous to basic ordering relations determined by the α-algorithm [79]. They also used the concept of *footprint matrix*, also known as *order matrix*. Their preliminary study was mainly aiming some further studies that propose a process-based goal scenario. This approach could be a basis to set up strategies allowing rapid dynamic reconfiguration to meet demanded requirements. This paper did not involve any standard goal-oriented language in conjunction with a process mining technique.

2.6.2 Intention Mining

As described in Section 2.1, process mining techniques mainly focus on event logs, discovered models, and predefined process models to extract process-related insights [1]. Hence, process mining has been growing into an activity-oriented approach [138].

A research direction called *intention mining* has however recently emerged. Intention mining shares objectives similar to process mining's but here the main goal is to discover *intentional process models*, beyond activity process models. Intentional process models consider the intentions underlying activities rather than the activities themselves. Aligned with goal modelling characteristics, the reasoning behind the activities is specifically addressed by intentional process models. This research field is developed by taking into account the notions of *intention* and *strategies* of the process enactment [97], [98], [99].

From intentional process modelling to intention mining

While *intention mining* is quite a young field, *intentional process modelling* (with MAP) as a basis for intention mining goes back to the late 90s. According to the trend of this research field presented by Epure *et al.* [139], many studies in intention-oriented process modelling proved that the fundamental nature of processes is mostly intentional and hence the process should be modelled from an intentional point of view. In the regular research trend of intentional process modelling, event logs have however been neglected. This neglect is the motivation for research in intention mining. The main challenge of these studies is to identify and formalize intentions from event logs [97], [98], [99], [104], [139].

Process mining versus intention mining

Khodabandelou *et al.* [99] compare two process mining and intention mining approaches by applying process discovery techniques to find a process model from a case study's dataset of traces and discuss the results. To this end, the α-algorithm [79] and the Strategies Miner algorithm [99] are applied in process mining and intention mining approaches, respectively.

Khodabandelou [97] has elaborated some aspects of intention mining and described its similarities to and differences from other fields of research such as process discovery approaches in process mining, goal-oriented approaches in goal modelling languages, machine learning approaches in process mining, and recommendation techniques. She emphasized that the process discovery approaches discover process model from sequences of actors' activities, whereas intention mining relies on the intentional aspect of the processes to discover the process model. The intentions are defined as the motivation to achieve the goals. The behaviours of users depend on their intentions and all of the intentions are not detectable. It is important that the intentions can be modelled at different levels of abstraction. A model of high-level intentions represents parent intentions and a low-level model shows sub-intentions. The fulfillment of a parent intention depends on the fulfillment of its sub-intentions.

Also, the objectives, representation languages, algorithms, and tools of both fields are briefly introduced by Khodabandelou *et al.* [99]. As described in Section 2.1.4, there

are many process mining algorithms and some concrete tools based on them, but the algorithms of intention mining are yet limited to the *hidden Markov Model* (HMM) method (described later). Furthermore, there is no intention mining tool that extracts intentional process models and infers goals from activities.

Language selection for intention mining activities

In order to choose a goal modelling notation to be used in intention mining, Khodabandelou *et al.* [99] compared three goal-modelling approaches (*i**, KAOS, and MAP). The approaches were compared in terms of their variability for goals, rigidness of task-decompositions, and their operational semantic of tasks. They rejected *i** because, "*i* is not designed to be a variable framework therefore, it does not afford a high level of flexibility*" [99]. However, this argument deserves to be challenged since *i** takes *intentional* and *strategic* views and supports representation of variability through means-ends relationships, enabling to perform both "upward" and "downward" strategy evaluation and to analyze the types of variability [51]. KAOS was also rejected because it has a rigid task-decomposition mechanism, and hence modelling complex intentional processes would be difficult. In addition to the aforementioned modelling language, Khodabandelou [97] has considered GRL and Tropos as modelling languages and did not choose them as they are based on *i** and thus were considered to have limitations similar to *i**'s.

MAP (introduced in Section 2.2.4) was chosen because this notation represents task and strategies in a labelled directed graph (aligned with the intention mining techniques) and allows formalizing flexible processes [97], [101].

Hidden Markov models as a basis for intention mining

One cannot easily understand HMM-based intention mining methods unless he/she has a good knowledge of HMM, e.g., as tutored by Rabiner [140]. Addressing this important required knowledge, an introduction to HMM is presented here as a prelude to its usage in the upcoming description of existing intention mining approaches.

A *hidden Markov Model* (HMM) is a statistical model that consists of two main components: *states* and *observations*. The state of the model in any given time generates an observation in that time. For example, in a meteorology context, states are the weather

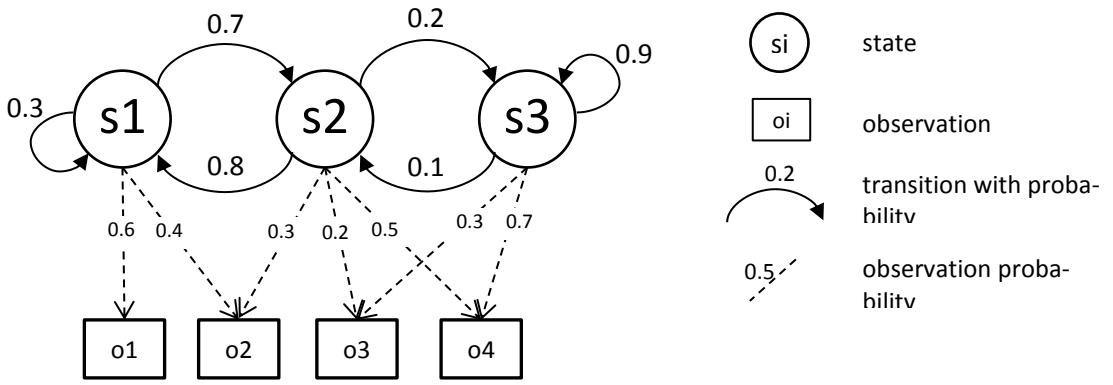


Figure 14. A hidden Markov model with three states and four observations

types (sunny, raining, etc.) and the possible observations generated with the states can be walking, shopping, staying at home, etc.

Figure 14 shows a simple HMM with three possible states and four possible observations. At a given time *t*, the state$_t$ is a member of the set of states called **S**. In Figure 14, **S**={*s1*, *s2*, *s3*}. State$_t$ depends on the previous state, at time *t*-1; the system goes from state$_i$ to state$_j$ with a probability called T_{ij}. Note that the sum of the probabilities of a given state's outgoing transitions is 1. For example, in Figure 14, T_{12}=0.7, T_{11}=0.3, and T_{13}=0 (0 probabilities are not shown in such models). Therefore, in state *s1*, the probability of moving to state *s2* is 0.7 while it is 0.3 to state *s1*.

Visiting a state at a given time can generate an *observation* from set **O**, with a given probability. In our example, the set of possible observations is **O**={*o1*, *o2*, *o3*, *o4*}. Let E_{ik} be the probability of generating observation *k* in given state *i*. In Figure 14, state *s1* can generate observations *o1* with a probability of 0.6 (E_{11}) and *o2* with a probability of 0.4 (E_{12}). Let the numbers of possible states and observations be *N* and *M*, respectively. Accordingly, there are two matrices that represent all T_{ij} and all E_{ik}, respectively called transition (**T**: $N \times N$) and emission (**E**: $N \times M$) matrices. For instance, the transition and emission matrices for the example in Figure 14 are:

$$\mathbf{T}=\begin{bmatrix} 0.3 & 0.7 & 0 \\ 0.8 & 0 & 0.2 \\ 0 & 0.1 & 0.9 \end{bmatrix} \quad \mathbf{E}=\begin{bmatrix} 0.6 & 0.4 & 0 & 0 \\ 0 & 0.3 & 0.2 & 0.5 \\ 0 & 0 & 0.3 & 0.7 \end{bmatrix}$$

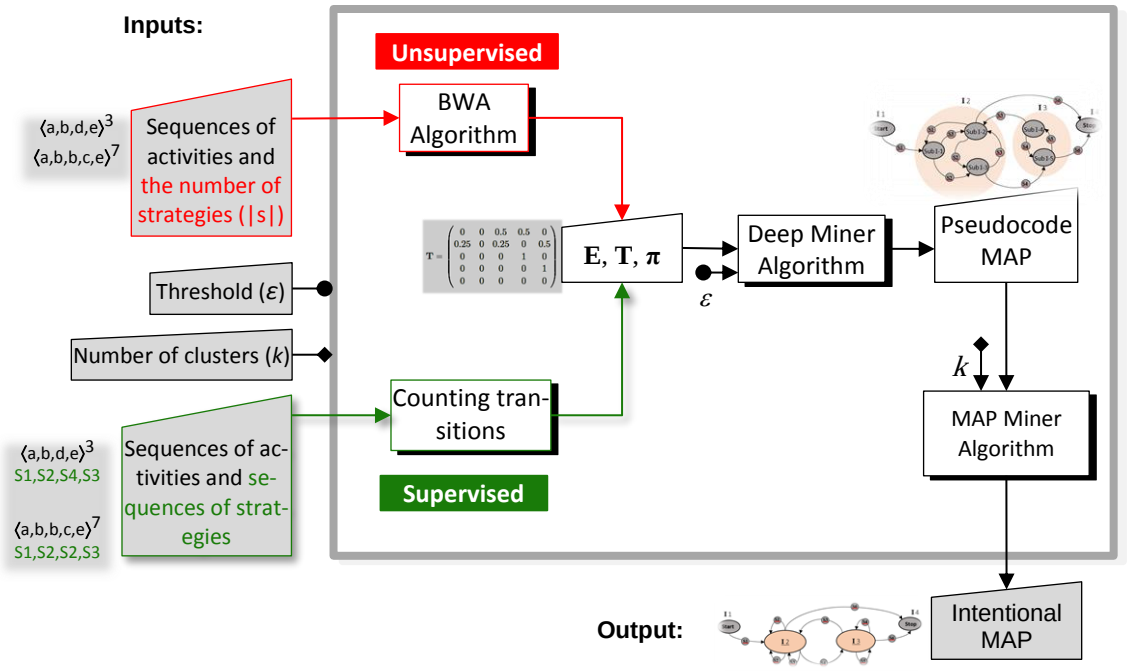


Figure 15. The supervised and unsupervised Intentional Map Miner Methods, with their inputs, algorithms, and outputs. The red items belong to the unsupervised method only, and the green items to the supervised method. The black items are part of both methods.

In addition, the vector $\boldsymbol{\pi}$ contains the probabilities of each state to be the initial state. In our example, $\boldsymbol{\pi}$=(0.9, 0.05, 0.05), which means that the system starts from *s1* with a probability of 0.9 (the $\boldsymbol{\pi}$ values are not shown in the figure). The transition and emission matrices, together with the initial vector, capture the entire model and can be formalized as a triplet, called $\lambda=(\mathbf{T}; \mathbf{E}; \boldsymbol{\pi})$. The main characteristic of HMM is that the states are not observable and remain hidden whereas the observations are the only things visible to users. Consequently, there are three key and pragmatic problems that should be solved, where *A* is a sequence of observations, e.g., *A*=(*o1*,*o1*,*o2*,*o4*,*o3*,*o3*):

- Problem 1: given the sequence *A* and the model $\lambda = (\mathbf{T};\mathbf{E};\boldsymbol{\pi})$, what is the probability of *A*, i.e., Pr(*A*| λ)?
- Problem 2: given the sequence *A*, what is the optimal sequence of states that could generate *A*?

- Problem 3: given the sequence A, how can we estimate the model parameters, $\lambda = (\mathbf{T}; \mathbf{E}; \boldsymbol{\pi})$, that maximize $\Pr(A|\lambda)$? Here, the transition matrix, the emission matrix, and the initial vector of probabilities need to be found.

There exist mathematical algorithms to solve the aforementioned problems. The *Forward Algorithm* is the main one for solving the firs problem. The second problem can be solved with the *Viterbi algorithm*, which finds the state sequence with the maximum likelihood, given the observations sequence. The third problem is however the most difficult one, as there is no known analytical way to find the most likelihood model. However, there is an iterative method, known as the *Baum-Welch Algorithm (BWA)* [141], which can find a local maximum likelihood estimate of the parameters given a set of sequences of observations [142].

Map Miner Method (MMM), unsupervised and supervised

The Map Miner Method (MMM) aims to infer a MAP intentional process model from event logs. If a prescribed MAP already exists, an analyst can compare the discovered MAP with the prescribed one and study the compliance level and deviations. In this sense, the MMM approach shares similarities with process discovery techniques in process mining. The output of process discovery is a discovered process model in the level of activity, while MMM ends up with a model in the intention level. Intentional MAP can be inferred through either an unsupervised or a supervised scheme. Here, we describe these two methods, which are illustrated in Figure 15.

In the work of Khodabandelou *et al.* [100], [102], [106] the focus is on *unsupervised* methods. They took advantage of HMM to propose a MMM framework to find an intentional MAP from event logs. In this framework, the hidden *states* are *strategies* and the *observations* are users' *activities.* Since activities are generated by users' strategies the activities that are observable in event logs would be generated by the strategy that the agent has adopted to achieve an intention. Therefore, the sequence of agents' strategies will generate the sequences of activities stored in the event log. For example, as Figure 16 shows, the trace of activities ⟨a,b,b,c,d⟩ observed in event logs could be generated by different traces of strategies such as ⟨s1,s1,s2,s2,s3⟩ or ⟨s1,s2,s2,s2,s3⟩.

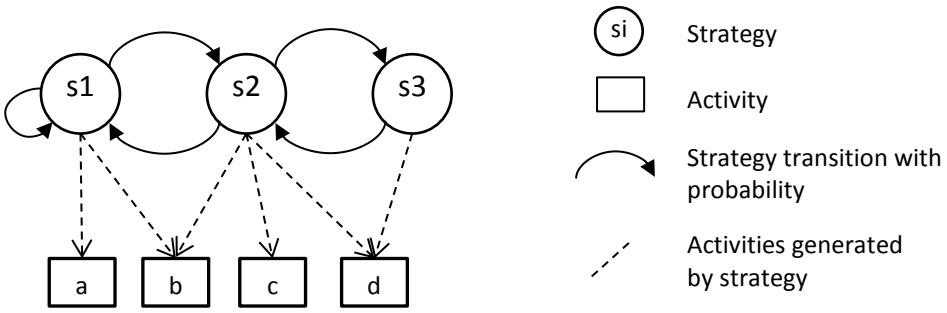


Figure 16. A sample hidden Markov model that consists of strategies as hidden states and of activities as observations.

Generally, each sequence of activities can be generated by different sequence of strategies with respect to the transition and emission matrices. As Figure 15 shows, in the unsupervised MAP Miner Method, the traces of activities are used as inputs. Although the strategies are some groups of activities, here, there is no previous knowledge about the strategies and the segmentation of activities' sequences into strategies. Therefore, in addition to the transition matrix (going from one strategy to another one), the emission matrix (generating an activity by a strategy) is also unknown. As the activities in the traces are not labelled by corresponding strategies, this method is called *unsupervised*. Here, the observations are known and the **T** and **E** matrices should be estimated. The HMM parameters are estimated using the Baum-Welch Algorithm (BWA) [141], the most frequently-used algorithm for learning the parameters of an HMM.

In addition to the observations, the BWA algorithm also needs the number of states as input. Therefore, the number of strategies should be defined by an expert in the unsupervised method. The main outputs of the BWA algorithm are the transition and emission matrices (see Figure 15). The transition matrix shows the probability of moving from one strategy at time *t* to another startegy at time *t*+1. Once **T** is estimated, the problem becomes how to extract a MAP process model (described in Section 2.2.4) that reflects the transition matrix well. The MAP extracted from a transition matrix should be verified in terms of two constraints: (i) *fitness:* any transition between strategies possible in the transition matrix should also be possible within the MAP; (ii) *precision*: any transition that is possible between strategies in the MAP should also be possible in the transition matrix. Deneckère *et*

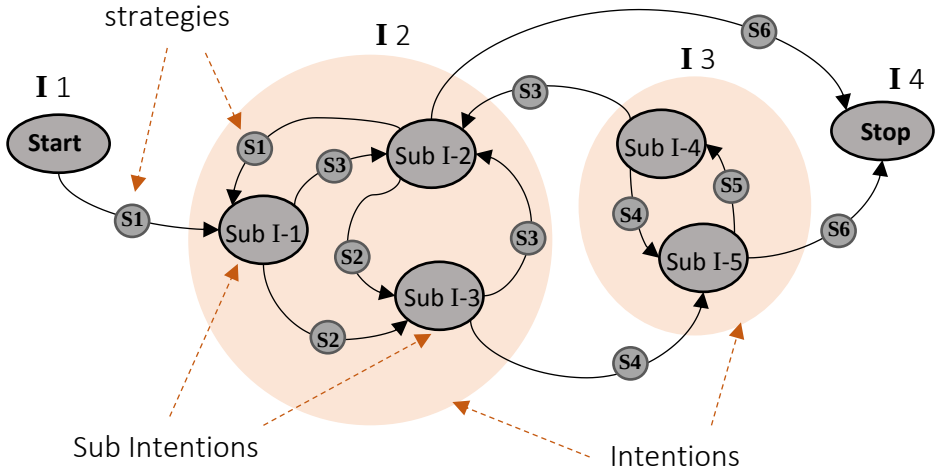


Figure 17. A fine-grained Map extracted by Deep Miner algorithm includes six strategies [108].

al. [101] and Khodabandelou *et al.* [100], [102], [106] introduced *recall* and *precision* as metrics of fitness and precision, respectively.

The main assumption here is that such unsupervised approaches only consider the transitions whose probability is above a given threshold ε. The value of ε is chosen heuristically to obtain a suitable trade-off between the granularity of the MAP and its understandability. Hence, ε is considered as an input of the algorithm in Figure 15. The F-measure metric, a combination of both recall and precision, is used to qualify the MAP. Deneckère *et al.* [101] proposed an algorithm, called *Deep Miner*, that finds a MAP that maximizes F-measures with the lowest number of sections (see Figure 15). The output of the Deep Miner algorithm is a very detailed map with several nodes, called *fine-grained* MAP. Figure 17 shows a *fine-grained* MAP extracted by the Deep Miner algorithm in an example with six strategies. The nodes generated by the Deep Miner algorithm are called "*sub-intentions*" as they are connected to each other with some strategies.

Deneckère *et al.* [101] and Khodabandelou *et al.* [104], [102], [100], [106] also proposed a MAP Miner algorithm to extract a high-level MAP from the fine-grained MAP produced by Deep Miner. To this end, the MAP Miner algorithm uses a clustering algorithm, *K-means* [143], to group the sub-intentions into some higher-level nodes (as clusters). In the clustering step, the attributes of each given sub-intention are defined as binary values that represent the connectivity of that sub-intention to the other nodes. Note

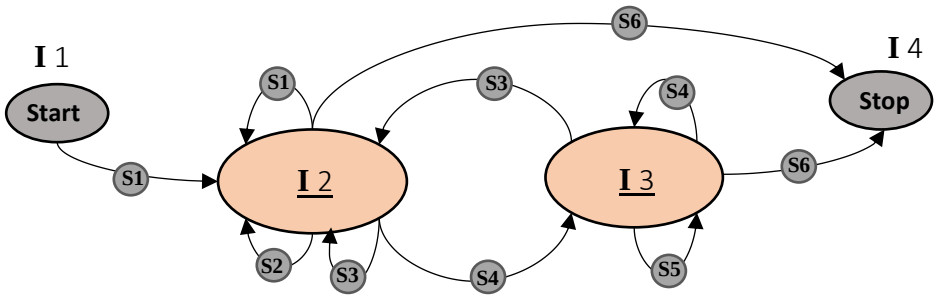


Figure 18. A coarse-grained MAP generated from the fine-grained MAP in Figure 17, using the MAP Miner algorithm [108].

that in the K-mean clustering algorithm, the number of clusters (*k*) has to be chosen in advance. Accordingly, the parameter *k*, which is the number of intentions in the final process model (coarse-grained MAP), is another input of MAP Miner. This allows determining the level of abstraction for the final intentional MAP. The MAP Miner algorithm groups the five sub-intentions of Figure 17 into two high-level intentions (I2 and I3), as shown in Figure 18.

In the supervised approach, proposed by Deneckère *et al.* [101], the steps are similar to the ones for the *unsupervised* approach, but the inputs are slightly different. In the supervised approach, the sequences of strategies (not just their number) are also known. For example, in the model shown in Figure 16, the sequence of activities (e.g., ⟨a,b,b,c,d⟩) and the corresponding sequence of strategies (e.g., ⟨s1,s1,s2,s2,s3⟩) are known.

In hidden Markov models, when the sequences of observations and their corresponding states are known, estimation of emission and transition matrices is simple. This problem, called HMM learning, is not one of the three aforementioned problems of HMMs. Here, nothing is really hidden. When the underlying states are known, a Maximum-Likelihood Estimation approach estimates **T** and **E** to have the most likelihood of generating the sequence of observations and the sequence of states simultaneously. In this approach (called *counting transitions method*), T_{ij} is the number of transitions from state *i* to state *j* divided by the number of times that state *i* occurs. Also, E_{ik} is the number of times that observation *k* is generated by state *i* divided by the number of times that state *i* occurs.

As shown in Figure 15, in a supervised approach, the BWA algorithm is not needed and the counting transitions method can simply be applied to estimate HMM parameters.

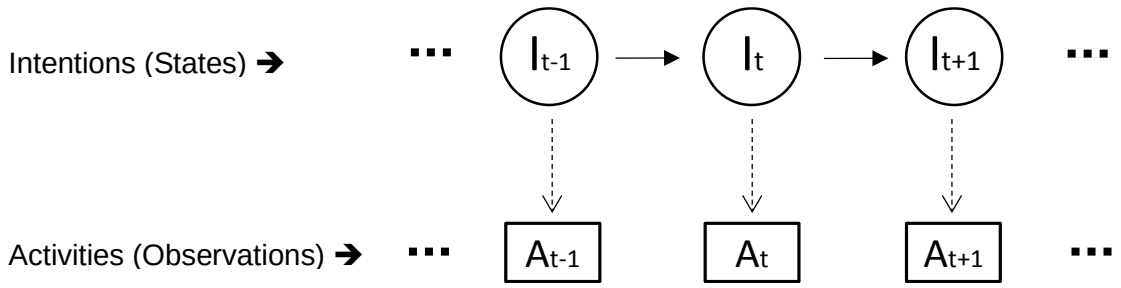


Figure 19. The HMM used in the method of Khodabandelou *et al.* [102]

After this point, all the steps are similar to the unsupervised approach and the MAP intentional model will be finally extracted. It is worth noting that the number of clusters (*k*) in the MAP Miner algorithm does not have to be chosen in a supervised approach as it equals the number of strategies that are known [98].

Comparing the performance of supervised and unsupervised approaches

Khodabandelou *et al.* [103] compared supervised and unsupervised learning approaches of HMMs in both a theoretical context (in terms of convergence speed and likelihood) and a practical context (in terms of the intentional process models obtained). The results showed that supervised learning ends up having a poor performance because it enforces binding conditions in terms of data labeling, introduces inherent human biases, and delivers unreliable results in the absence of ground truth. On its side, unsupervised learning gets efficient MAPs with a better performance and lower human effort. Rabiner & Juang [142] also demonstrated that an unsupervised learning approach to estimate HMM parameters offers a higher performance than supervised learning.

Supervised intentional process model discovery using MMHs

Khodabandelou *et al.* [98] proposed an intention mining method called supervised intentional process model discovery using MMHs. This approach is similar to the supervised MAP Miner Method, but here the states of the hidden Markov model are *intentions* instead of *strategies* (see Figure 19). This approach looks for some predefined intentions behind the user activities traces and compares them to a prescribed intentional model that is represented with MAP.

In this method, there is a prior model called *intentional process model*. This model is a MAP that consists of some strategies and some intentions. A human expert describes this model. Users can select the alternative strategies to fulfill their intentions, following this intentional model. Then, a modeller makes some groups of activities based on his/her inference to define some strategies that are shown in the prescribed intentional model. Similar to the supervised MAP Mining Method, this approach adopts hidden Markov models to discover a MAP from observations. Consequently, regarding the characteristics of HMMs, a sequence of activities (traces) of a given length could be generated by many sequences of intentions with the same length. Here, the purpose is to find one sequence of intentions among all possible ones that is the most likely sequence of intentions, which generates the given sequence of activities. This problem is exactly the second key problem of hidden Markov models described above.

The *Viterbi algorithm* (VA) [144] is a well-known algorithm for finding the most likely sequence of hidden states that generates a sequence of observed events in hidden Markov models. This algorithm needs transition and emission matrices. When providing these inputs, the modeller estimates the transition matrix by the simple counting approach described before (in the supervised MAP mining approach). Once the parameters of the HMM are estimated, they can be used to find the hidden intentions behind any sequence of actions traces (the sequence with the highest probability of emergence), using VA. Then, this discovered sequence of intentions could be compared to the prescribed sequences of intentions by a modeller. At this point, Deneckère *et al.* [101] realized that this method is not accurate enough to represent a MAP model. They argued that activities are indeed generated by strategies, not directly by intentions. Accordingly, Deneckère *et al.* [101] proposed their supervised Map Miner Method, which models hidden states as strategies and observations as activities. Note that the supervised Map Miner Method infers high-level intentions by clustering the sub-intentions extracted from the Deep Miner algorithm (shown in Figure 18).

Unsupervised intentional process model discovery using IntentMiner

Based on the intention mining approach proposed by Khodabandelou *et al.* [98], Epure *et al.* [105] proposed an approach, called *FlexPAISSeer*, that focuses on the difficulties of

unexperienced actors when *enacting* flexible processes. An unexperienced process actor, who is not much aware of the process, may have some difficulties to make a good decision about the action to execute next under specific constraints. Addressing this problem, *FlexPAISSeer* offers two components: *IntentMiner* and *IntentRecommender*. The first one discovers the intentional model in an unsupervised manner, then the second one makes recommendations based on the discovered intentional model and probabilistic calculus. In this unsupervised approach, the clusters of events that are associated with an intention are identified as a basis for the intentional model.

Epure *et al.* [105] aim to discover the intentional model through the traces of the process participants. To this end, they mine both the intentions and the flows between them. They first focus on the identification of the relevant data in a specific structure. Then, they follow the *IntentMiner* algorithm and its six steps: extract the process participant current log, sort the log by time, apply syntactic analysis, apply trend analysis, apply semantic analysis, and aggregate intentions. While Khodabandelou *et al.* [38, 40] used BWA to find clusters of the strategies, Epure *et al.* [105] proposed an algorithm to cluster the events in some classes and analyze them as intentions. The main idea of their clustering is based on the similarity and confidence between the consequent events that are sorted by time. The attributes of events are the main basis for finding the correlation between them. Once the intentional clusters are determined, the next step is the intention mining and naming by applying semantic analysis. Finally, the algorithm aggregates the mined process instances in the intentional process model. The intentional process model is further used by Intent-Recommender for providing up-to-date recommendations to the process participants.

Epure *et al.* [105] evaluated IntentMiner in an experiment with 10 participants, interacting with a Childcare system developed by a software company and used by several child day care centers in The Netherlands. They compared the intentions classified by IntentMiner with the real intention of participants. Following a confusion matrix classification, the average precision was 69% and recall was 97% (IntentMiner mined the process participants' intentions in 97% of cases). Khodabandelou *et al.* [98] reported an average precision of 97% and average recall of 93% for their supervised intention mining technique based on Hidden Markov Models. The precision was much better given the fact the classifier had been trained earlier.

2.6.3 Key Performance Indicators (KPIs)

Based on the definition of Flapper *et al.* [145], performance is the degree to which an organization accomplishes its objectives. All performance indicators (PIs) should be linked to the organization's goals [145]. The PIs that reflect the critical or "key" success factors of the organization and are used to define and measure progress towards the organizational goals are called KPIs [145], [146]. The fact that KPIs are known means of monitoring goals is an important motivation to take KPIs into consideration in this literature review. The studies that focus on the use of event logs in monitoring the KPIs are worthy of attention.

As discussed earlier, the analysis of misalignments between an event log and a prescribed process model is a good source of improvement of requirements. Process mining techniques (i.e., conformance checking and process enhancement) make it possible to diagnose such deviations. The severity of each deviation can be *quantified* to form a basis for tackling the problem of repairing a process model such that the enhanced model can replay the log (as a history of actual behaviours) [147], [148]. Since a process model may deviate from a log at an arbitrary number of points, making such alignment maybe very far from trivial, van der Aalst *et al.* [148] have demonstrated characteristics, limitations, and methods for conducting conformance checking in the context of process mining. In the same context, Fahland and van der Aalst [134] have investigated the problem of repairing a process model to replay the log and conform to it.

Relying on [147], [148], Dees *et al.* [109] proposed a methodology that repairs a process model with respect to the behaviours that do not violate any rule and have a significant improvement on the performance level. Involving performance levels that correspond to specific business goals or requirements, represented by KPIs, positions their work as a goal-oriented process mining activity. The basic inputs of the proposed methodology are an event log together with a process model (either discovered or designed manually) represented on Petri net. The output is a repaired and improved process model ready to impose a better way of performing the business process.

In a nutshell, Dees *et al.*'s methodology is implemented in three main steps. First, a deviation analysis is performed to determine which deviations have a positive impact on the process performance. To this end, deviations are correlated to a selected KPI stored as

a field in event log. Then a set of rules is discovered through a decision tree classification method, where the KPI is a label field. Then for each rule, the corresponding class contains all traces that comply with the rule. In the second step, traces of different classes are repaired to only keep those deviations having a positive impact on the KPI. Then all the logs kept in all the classes are merged to obtain a single repaired event log. Finally, the repaired log is used as a basis to repair the process model. To this end, the process model is revised in such a way that the resulting model is able to replay all the behaviour of the repaired event log. In particular, this technique will modify the model only to make the desired deviating behaviours possible. Here, the desired deviating behaviours can be considered as those facts that can reflect some neglected requirements.

Dees *et al.* [109] assessed the feasibility of their proposed methodology through two case studies. The result showed that the improved process model still adheres to the desired prescriptive model and guarantees enhanced KPI levels.

While Dees *et al.* [109] took advantage of performance indicators to improve business processes, Cho *et al.* [110] aimed to assess whether process redesign have been applied and to what extent. They proposed an assessment framework, where the focus is on the process redesign lifecycle phase, which is tightly coupled with process mining as an operational framework to calculate identified indicators.

The framework consists of some indicators to assess whether best practices have been applied. Best practice implementation indicators (BPIs) and process performance indicators (PPIs) are two sets of indicators that assess process improvement resulting from the application of best practices. It is possible to calculate both sets using standard process mining functionalities. Cho *et al.* also define what data should be recorded during process execution to enable such a calculation. The proposed framework was evaluated over case studies in a hospital and a tour agency and compared with other approaches in the literature.

Cho *et al.* [110], defined 17 BPIs for 29 best practices that were previously proposed by Reijers and Mansar [149], such as *customer reduction* and *customer integration*. They also suggested process mining techniques appropriate for computing each indicator. In the case where the BPI information does not exist in the event log, they suggested which supplementary information is needed to assess the implementation of redesigns. In addition, 13 PPIs are suggested to assess the effect of best practices on process performance.

These PPIs are based on four process performance measures explained by Reijers and Mansar [149]: Time, Cost, Quality and Flexibility. Cho *et al.* gave a detailed explanation on each PPIs including how to measure them using event logs. Most business process redesign efforts are invested in increasing the efficiency of business processes through improving time-related indicators, e.g., decreasing processing time and waiting time. Calculating time-related PPIs (5 out of their 13 PPIs) using event logs is much easier than calculating quality-related indicators (4 out of their 13 PPIs) because they are not clearly stated in the logs. The latter are rather evaluated by checking the satisfaction of customers, for instance via surveys [150].

Krathu *et al.* [108] used a bottom-up approach for the identification of KPIs from event logs, business information, and process models. This approach, together with a top-down approach for measuring business performance on the strategic level (Balanced Scorecard, BSC), make a framework that supports inter-organizational performance evaluation. This framework, proposed mainly for evaluation of inter-organizational relationships, is developed as a plug-in of ProM 6, namely the BSC EDImine plug-in [151].

Knowledge about the BSC method is essential to understand the structure of this framework. In terms of the bottom-up approach that uses event logs for identification of KPIs, they follow two major steps: i) *frequency analysis* and ii) *consideration of KPIs based on the semantics of data elements and message types*. This method has been described in detail by Krathu *et al.* [107]. It is worth noting that these two papers [107], [108] represented the EDImine plug-in, specifically designed for supporting the business activities and transactions within a network of organizations called Electronic Data Interchange (EDI).

The framework proposed by Krathu *et al.* [107], [108] was validated with a case study using data from a manufacturing company. This case study showed that the company can evaluate the inter-organizational performance quantitatively against their business objectives. The KPIs derived from this bottom-up process mining techniques together with other statistical techniques enabled the company to evaluate their business performance more accurately.

2.7. Threats to the Validity of the Literature Reviews

The goal of a systematic review is to uncover as many primary relevant studies as possible to summarise all current information about some phenomenon in a systematic and unbiased manner. There are several threats to the validity of our research and of the papers that were reviewed in the current chapter. Validity of a research is concerned with the question about the correctness of conclusions, i.e., the alignment between conclusions and reality [152].

In this section, we consider the threats and the biases that could affect the validity of our work and of the papers selected for review. The main threats are discussed according to two categories: construct validity and internal validity [153].

2.7.1 Construct Validity

Construct validity refers to the quality of the methodology in terms of being helpful to answer the target research questions. Here, searching for and selecting relevant papers played an essential role. Note that our first question (RQ1_SLR1) in Section 2.3 was directly answered based on counting the papers that are related to process mining, while the two other questions (RQ2_SLR1 and RQ3_SLR1) were answered based on some papers selected from the set found during the searching stage. Therefore, any weakness and inaccuracy in the first stage could threaten the correctness of our answers. Also, regarding the papers reviewed in Section 2.4, all but three papers ([65], [66], [70]) have clearly used search engines to provide their paper sets. As these papers are literature reviews, their research questions are answered based on the results of their queries. Therefore, their validity level depends on their own choice of queries, databases, and search engines. Unfortunately, only Breuker and Matzner [63] have reported the steps of their search, including search engines (Google Scholar and Scopus), keywords and inclusion and exclusion criteria. Here, two categories of threats that are associated with selecting papers are discussed.

Missing some relevant references: Although we included the most frequent and essential keywords in the queries, there is a possibility that some relevant papers have not been found. For example, the very first paper in the context of process mining (Cook, 1996), was not found by our queries of Section 2.3, as it focused on “process discovery” rather than on process mining. Breuker and Matzner [63] also used “process mining” in their query.

The second threat is related to the search engine limitations. For example, Google Scholar only gives the possibility of search within title or full-text, we opted to search in titles only (to avoid noise). Consequently, the papers that do not have the queries' keywords in their titles were not detected by Google Scholar.

Given that we have defined our queries' keywords in English, another threat that may affect the validity of our review (and the eleven papers we reviewed) is the likelihood of ignoring some relevant non-English papers.

Taking advantage of four search engines, including two major ones (Google Scholar and Scopus), enables us and Breuker and Matzner [63] to detect a large proportion of the papers stored in scientific databases. However, there might be some relevant papers stored in other databases that are not accessible with these search engines.

Finally, our first literature review was completed in early 2016, and so more recent papers are currently missing (although they would unlikely change our answers to the related research questions).

Collecting irrelevant references: The threats in this category are related to finding irrelevant papers through the queries. As described in Section 2.3.1, we iteratively used more constraints to filter out some irrelevant papers, specifically the ones related to the mining or chemical industry. Yet, reviewing every single paper is the only way to make sure that there is no remaining irrelevant paper. We performed this approach on the 36 selected papers. Breuker and Matzner [63] have also used some automatic filtering to exclude some papers. They have found 2016 papers for process mining and 5693 for organizational routines.

One of the important parts of a systematic literature review is to clarify the study selection criteria. These criteria are intended to identify the relevant studies that convey direct evidence related to the research question while avoiding bias [64]. Regarding the papers considered in Section 2.4, as seen from Table 4, only two ([63] and [69]) have clarified their selection criteria.

According to these two kinds of threats, there is a chance that our trend-graphs may not be precise (RQ1_SLR1 in Section 2.3.1). Also, the selected papers may not capture all papers that are published related to literature reviews in process mining in general or in goal-oriented process mining in particular.

Internal Validity

Some other biases and confusing factors can threaten validity of our study and the studies that we reviewed. For example, Maita *et al.* [62] focused on the use of artificial neural networks and support vector machines in process mining. They reviewed 11 papers and concluded that the attention that has been paid to ANNs and SVMs by process mining researchers is less than expected. Then, they related this to process mining researchers' ignorance of these two data mining techniques. These two conclusions could be wrong because of the bias of the authors.

In this research, one additional threat is that the selection of papers and the application of the criteria was mainly done by one person, with a confirmation from the supervisor. We suspect the other eleven papers to be in a similar situation (but such threats are rarely discussed in their papers). Such issue could be mitigated by having more people do such selection independently, with a consensus mechanism.

2.8. Data Preprocessing in Process Mining

Though sophisticated in detail, goal-oriented process miming and our GoPED algorithms are, in essence, a filtering step added to typical data preprocessing before process mining algorithms are applied. The main output of GoPED is a subset of the original event log that contains the event information of some cases selected with regards to some goal-related criteria. This subset will be fed to process discovery applications to generate a model promising to meet the predefined goal-related criteria. In this section, the current state of data preprocessing in process mining is considered to reveal the position of goals in it and highlight the filtering techniques currently used in process mining.

2.8.1 Importance of Data Preprocessing

Process mining tools adopt several approaches to simplify complex event logs and mitigate the generation of spaghetti-like models: *abstraction*, *event log filtering*, and *trace clustering* [154]. Abstraction is not applied directly on the event logs; instead, it is used to remove a subset of the nodes from the process map, to reduce a smaller dependency graph of the given event log (therefore, its detail is out of the scope of the current section). Abstraction

techniques are often helpful because they allow the analysts to aggregate activities/paths of spaghetti models and have a better understanding of the process on a macro level. Nevertheless, although the abstraction approach can help visualize large event logs, it is not efficient enough to cope with the full complexity of real event logs [76]. Accordingly, process mining research emphasizes *event log filtering* [155] and *trace clustering* [84, 156] as promising types of event log simplification. While filtering techniques work well with structured processes, they have problems discovering less structured ones [155]. Trace clustering [84, 156] has been developed to cope with unstructured models.

2.8.2 Event logs Filtering

Preprocessing event logs before applying a process mining algorithm is a de-facto practise that typically includes a log filtering phase [155]. Augusto *et al.* [157], in their systematic review of automated process discovery methods, showed that for complex event logs, it is highly recommended to use filtering of the logs before process discovery. Without such filtering, the precision of the discovered models becomes very low.

Common event log filtering approaches are performed using three categories of filters that remove *i*) a subset of the *traces*, *ii*) a subset of the *events*, or *iii*) a subset of the *event pairs* to produce a simpler log.

Trace Filters

This type of filter can be used to keep or remove all the traces that satisfy a pre-defined condition. For example, all the traces that happen with a pre-defined range of frequency, or all traces whose duration of cycle time is in a specific range can be filtered out.

Event Filters

Using the event filters, the modeller can remove or keep all the events that conform to a pre-defined condition. If, as a result, all events of a trace are removed, then the whole trace will be removed (i.e., no empty traces remain).

Event Pair Filters

This category of filters enables the modeller to remove or keep all the *pairs* of events that satisfy a specific condition. This type of filtering concerns relations between two events

and gather more insight into temporal relations of activities. For example, one can filter out all event pairs where event *a* is directly followed by event *b*.

Filtering in ProM

The ProM framework, as the most popular toolkit used by process mining researchers, offers several plugins for event log filtering [158]. The majority of research activities in process mining have an accompanying implementation in ProM. Most research work on general-purpose event log filtering considers ad-hoc filtering implementations within ProM. Many of these implementations are beneficial when the user aims to use a specific subset of traces/events instead of the whole event log [159]. The event log filtering plugins implemented in ProM are as follows. All these plugins get the whole event log as input and obtain a subset of the event log (i.e., filtered event log) as output.

- ***Filter Log using Simple Heuristics (SLF) plugin***: This plugin removes traces that do not start and/or end with a specific event, based on its frequency, e.g., filtering out all traces that start with an infrequent activity/event. This plugin is mainly used to filter out incomplete traces [160].
- ***Filter Log on Event/Trace Attributes Plugin***: This plugin removes events or traces based on a certain attribute. All traces/events whose specific attribute value does not meet a user defined condition will be removed [159].
- ***Filter Out Low-Frequency Traces plugin***: This plugin keeps a trace only if its frequency is not less than a user-defined threshold. This plugin is mainly used to filter out non-frequent traces [161].
- ***Filter on Timeframe plugin***: This plugin concerns only the timestamp. The events occurred within a user-defined time window (e.g., the first year of recordings) will be saved and all other will be filtered out.
- ***Filter Log using Prefix-Closed Language (PCL) plugin***: This plugin removes events from traces to make an event log that can be represented as a Prefix-Closed Language [162]. A language *L* (set of some strings/words) is prefix-closed if, for all words in *L*, every prefix of the word is also in *L* [162]. This plugin removes all traces that are not a prefix of another trace in the log based on a frequency threshold defined by the user.

It is noteworthy that the filters and the filtering schemes explained above might are implemented by processes mining researchers in multiple packages, some of which available in the last version of ProM. The details of configuration of filtering packages are explained in ProM's documentation [163].

All plugins discussed above require some form of domain knowledge to work properly. Furthermore, typically the modeller should set one or more (complex) settings to configure the plugging [159].

2.8.3 Trace Clustering

The *trace clustering* technique splits the event log into homogeneous subsets, and then the subsets are used to produce separate process models [164, 165, 166]. This approach can cope with real flexible environments (like healthcare) and improve process mining results [156]. The main idea behind the traditional trace clustering technique is to group traces that share similar features. The clustering techniques used by the process mining community come from the traditional clustering techniques used and developed by the data mining community. Therefore, in the clustering stage, clusters are formed and evaluated based on the data mining idea of maximizing similarity within a group and minimizing similarity between groups. The traditional clustering techniques apply either partitional techniques like K-means or a hierarchical method for generating clusters.

Although clustering techniques have proven useful, they suffer from a fundamental problem rooted in the difference between what is concerned with clustering methods and what process discovery schemes consider. Clustering techniques take into account only the similarity between objects and proximity between clusters to create clusters. Therefore, the performance of the clustering is evaluated using some measures (e.g., Davies–Bouldin index, and Silhouette Coefficient [167]) that mainly consider the distance between the objects. As such, the traditional event log clustering techniques do not take into consideration the quality of the process models associated with each event cluster during the clustering procedure. Therefore, these techniques suffer from a strong divergence between the clustering bias and the evaluation bias. Accordingly, it is highly questionable whether the traditional clustering techniques will result in acceptable results in terms of process models [156].

The principal evaluation measurement for a trace clustering technique is from a process discovery perspective. This requires that for each cluster, a process model is created by a certain discovery technique and the quality of a certain clustering solution is determined by combining the process model accuracy and process model complexity [168].

Solving the bias divergence of traditional clustering approach, *Active Trace Clustering* [156], as an entirely different approach inspired by *active learning,* was developed. The main idea behind this approach is to find an optimal distribution of traces among a given number of clusters whereby the combined accuracy of the corresponding process models is maximized. *Active Trace Clustering* borrows elements from machine learning and uses a selective sampling strategy that enables an active learner to decide about the selection of instances based on their informativeness. The *frequency of the trace* is the most common factor used as an informativeness measure.

There is, also, an approach in trace clustering that uses Levenshtein distance, also called as edit distance, to measure (dis-)similarity between two sequences/traces [166]. In this approach, the cost or weight of edit operations (*insertion*, *deletion*, or *substitution* of an element) is determined by taking into account the context of an event within a sequence/trace [166]. In this approach, for example, the substitution of uncorrelated activities is discouraged, and the substitution of contrasting activities is penalized.

Trace Clustering in ProM

The ProM framework offers several ways of clustering similar cases on the basis of selected features. Typical techniques like K-means clustering [143] can be employed as a preprocessing step for process mining. It is noteworthy that considering the clusters themselves may result in novel insights about the event log and the underlying processes. Furthermore, the clusters can be used to discover several simple process models instead of a single complex one. Below are two clustering plugins implemented in ProM [169].

- ***Disjunctive Workflow Schema (DWS) mining Plugin***: According to the approach of this plugin [164], first, the whole event log is used to discover a model using the HeuristicsMiner algorithm [77]. Then, if the discovered model is optimal without any over-generalization issue, the approach stops, otherwise the log is split into clusters using

the K-means method. Then the plugin discovers models for each cluster. If such models are not yet optimal, the clusters are repartitioned and so on until optimal models are achieved.

- ***Trace Clustering***: This plugin aims to partition the event log by grouping similar sequences together [84]. This approach uses distance-based clustering accompanied by profiles to lower the number of cases analyzed at once and generate models with lower diversity and complexity. Profiles are composed of a set of features that describe and classify cases from a particular perspective. Then, similar cases are grouped into the same clusters using clustering methods such as K-means clustering or Self-Organizing Maps (SOM).

2.8.4 GoPED as a Novel Event Log Filtering Approach

As explained above, the existing data preprocessing approaches for process mining rely on filtering events based on their frequency and attributes in the layer of activities. Similarly, the clustering approaches take into consideration the similarity of traces at such a level. The goal-related attributes and the factors at the level of intention and goals are not considered, neither to filter event logs nor to cluster or abstract them. In goal-oriented process mining, this gap is bridged and GoPED exploits the goal-related factors and offers an event log filtering approach entirely different from existing ones.

2.9. Chapter Summary

This chapter provided an overview of the main concepts of the research, namely process mining and goal modelling. Then, it provided a literature review of process mining in general and of goal-oriented process mining in particular. Following the systematic literature review approach, we ran queries using four relevant search engines and analyzed the returned papers. The first review selected 11 relevant papers and in the second one 24 papers were selected. An informal review of existing approaches for minimizing trace logs, mainly based on filtering and clustering, additionally highlighted the absence of consideration for goals.

According to these reviews, focusing on goal-oriented process mining as a new line of research could make a beneficial connection from event logs, as a major part of the process mining concept, to goals and requirements, as the main concepts of most requirements engineering approaches. This connection opens new horizons in the context of business process modelling and data analysis to mine, monitor, and maintain process goals, process performance, and satisfaction levels of stakeholders. This research direction, which would fill the *goal-oriented process mining* gap highlighted in this chapter, could lead to developments that would help practitioners understand and improve processes, as well as support organizations in better measuring and achieving their goals.

The next chapter will explain the research method used in this thesis, which exploits the knowledge generated through the above literature reviews.

Chapter 3. Research Method

This chapter explains how Design Science Research, and especially the version for information systems presented by Peffers *et al.* [5], is used to conduct this thesis research. To this end, the fundamentals (i.e., activities) of such a version of DSR will be briefly presented, and their usage in this thesis will be detailed.

3.1. DSR Activities and Concepts

Over the past 17 years, many versions and variants of the original design science (DS) research methodology have been proposed for different research contexts [170]. The version of the DS research methodology presented by Peffers *et al.* [5] incorporates principles, practices, and procedures required to perform research in an IS discipline oriented to creating successful artifacts. That version is consistent with prior literature, provides a nominal process model for doing DS research, and provides a mental model for presenting and evaluating DS research in IS [5, 171].

As Figure 20 shows, this DS process consists of six main six activities in a nominal sequence: 1) *problem identification and motivation*, 2) *definition of the objectives for a solution*, 3) *design and development*, 4) *demonstration*, 5) *evaluation*, and 6) *communication*.

3.1.1 Activity 1: Problem Identification and Motivation

In this activity, the specific research problem is defined, and the value of a solution is justified. The problem definition will be the main basis to develop an artifact that can provide a solution. Justifying the value of a solution motivates researchers to pursue the solution and accept the results and helps understand the reasoning associated with the problem. Knowledge of the state of the problem and the importance of its solution are the resources required for this activity.

In the thesis, it was revealed that the current process mining approaches are mainly formed at an activity level, and the goals and their satisfaction levels are not taken into

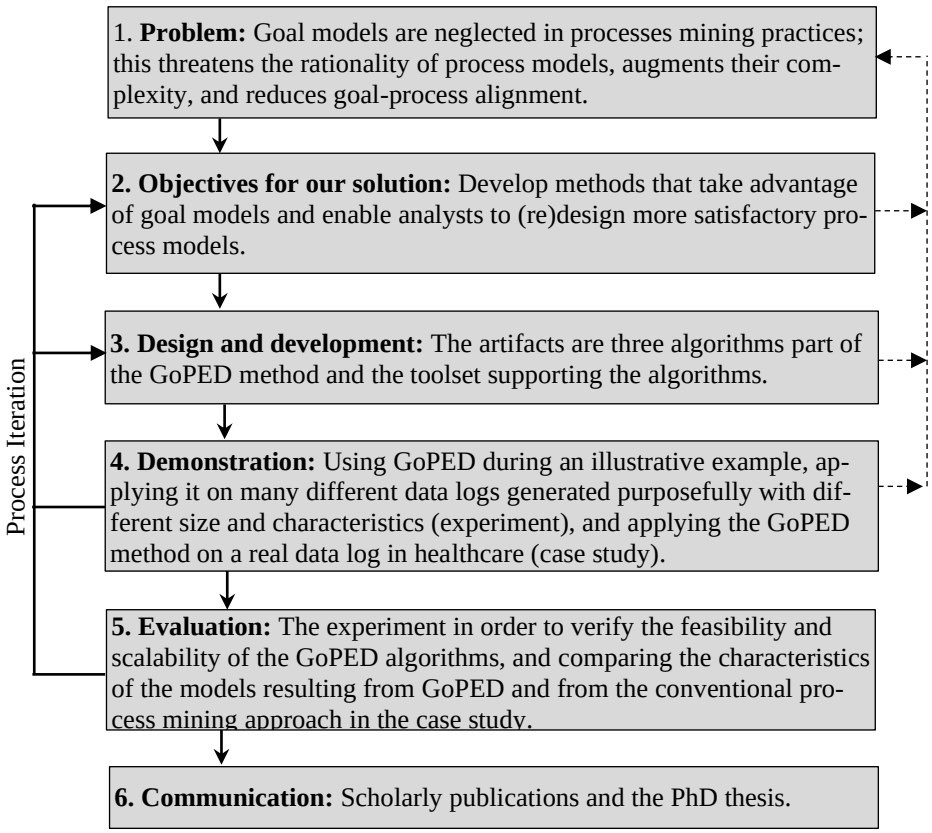


Figure 20. Research method, instantiating DSR [16]

consideration. We recognized that such neglection results in unreliable rationality behind the discovered models and unstructured "spaghetti-like" processes. Finding a way that makes the discovered models aligned with stakeholders' goals on the one hand and results in models easy to understand and analyze, on the other hand, will be of interest to process mining researchers and practitioners.

After three literature reviews explained in Chapter 2, we have the knowledge of the state of the defined problem and the importance of its solution as the resources required for problem identification.

3.1.2 Activity 2: Define the Objectives for a Solution

In this activity, the objectives of a solution of the defined problem and knowledge of what is possible and feasible is inferred. Such objectives should be inferred rationally from the problem specification and can be quantitative or qualitative. Knowledge of the state of

problems and existing solutions, if any, and their efficiency are the resources required for this activity.

Based on the problem identified by this thesis, the objective for the solution was determined. We aim to develop a method to enrich process mining activities such that they exploit both event logs and measurable goals (derived from goal models) in order to generate models that are both aligned with goals and easy to analyze. Such a method would allow process mining researchers to use the goal models and their satisfaction levels- together with the typical attributes of an event log to discover models. Resulting process models will satisfy the goals and enable the analysts to analyze the resulting models and gain some insight about the processes and improve them.

The second systematic literature review that focuses on goal-oriented process mining (reported in Section 2.5) and the review of current data preprocessing techniques used for process mining (reported in Section 2.52.8) revealed the state of the defined problem and existing solutions. We found that these solutions are rarely available, and that a solution that addresses the research problem comprehensively does not exist. Hence, we have the knowledge of existing solutions and their efficiency as the resources required for this activity.

3.1.3 Activity 3: Design and Development

The artifacts, which are potentially constructs, methods, models, or instantiations (each defined broadly in [172]), are created in this activity. *Constructs* offer a language in which problems and their solutions are defined and communicated. *Models*, however, use constructs to represent a real-world situation that is the design problem and solution space. Models help analysts and researchers understand problem and solution and represent the connection between problem and solution components. They enable exploration of the effects of design decisions and changes in the real world. *Methods* offer a guide on how to solve the defined problems. These can range from formal, mathematical algorithms to informal, textual descriptions of "best practice" approaches or their combination. *Instantiations* show that the artifacts (constructs, models, or methods) can be implemented in a working system. Instantiations allow concrete assessment of an artifact's suitability against

its intended purpose to demonstrate its feasibility. They also aid researchers learn how the artifact affects the real world.

The main artifact of this thesis that directly addresses the defined problem is GoPED (explained in Chapter 4 and Chapter 5). GoPED can be considered a *Method* artifact as it consists of three algorithms to find process models aligned with goals from three different perspectives. Such artifact can also be labelled as a *prescriptive* DSR knowledge as it is characterized as "*how*" knowledge, while *descriptive* knowledge is the "*what*" knowledge about phenomena [173].

In the "design and development" activity, the artifact's desired functionality and its architecture are determined, and then the actual artifact itself is created. Accordingly, after designing the algorithms, they were implemented in a GoPED tool, supported by three other tools used to pre-process inputs and post-process outputs.

Another artifact of this thesis is the descriptive knowledge about the state of process mining in general and goal-oriented process mining, in particular, provided through two systematic literature reviews described in Chapter 2.

The GoPED method is also instantiated and used on synthetic event logs in Chapter 6 and Chapter 7, and on a real-world case study in Chapter 8. Such instantiations allow assessing GoPED's suitability against its intended purpose as well as demonstrating its feasibility and scalability. The case study in Chapter 8 can also help researchers learn how GoPED can affect the real world. Hence, the illustrative example, the experiments, and the case study can be considered *instantiation* artifacts of this thesis.

3.1.4 Activity 4: Demonstration

The use of the artifact to solve the problems is demonstrated in this activity. Such a demonstration can involve the use of the artifact in experimentation, case study, simulation, proof, or other proper activities. Effective knowledge of how to use the artifact to solve the problem is required for the demonstration.

In the thesis, the use of GoPED to discover models aligned with predefined goals and to allow analysis that can provide some insight about process models is demonstrated in three chapters. Chapter 6 uses a *diagnosis of gestational diabetes* (DGD) process provided by a hospital located in Ontario, Canada, to illustrate the three algorithms of GoPED.

Then, the models resulting from GoPED are compared to the models resulting from the original event log, and the positive impact of GoPED is highlighted. Moreover, in Chapter 7, the GoPED method is applied to 1960 different event logs generated with a planned scheme to demonstrate the scalability and feasibility of the artifact against a wide variety of event logs with different sizes and configurations. Finally, Chapter 8 demonstrates the impact of the use of the artifact to find models aligned with goals in a real-world event log (case study involving a sepsis management process in a Dutch hospital).

The author who proposed the artifact and developed it is well-positioned to use it and has a solid knowledge of how to use it to solve the problem as required knowledge for the demonstration. However, for the case study where the goal-related attributes (as a required component of GoPED) were not explicitly available, and where the author did not have the medical knowledge required to derive the attributes from the event log, a physician was involved in confirming the goal modelling component of GoPED.

3.1.5 Activity 5: Evaluation

This activity observes and measures how well the artifact supports a solution to the defined problem. To this end, the defined objectives of a solution are compared to the actual results of the use of the artifact in the demonstration activity. Knowledge of analysis techniques and relevant metrics are required in this activity. Based on the nature of the problem and the developed artifact, the evaluation can take many different forms. Comparison of the artifact's functionality with the solution objectives, clients' feedback, satisfaction surveys, or simulations could be involved in the evaluation. This activity could also include quantifiable measures of performance, such as response time. Conceptually, any appropriate empirical evidence or logical proof might be included in such evaluation. At the end of the evaluation activity, the researchers can decide whether to iterate back to the activity of "design and development" to improve the artifact's effectiveness or to continue to the communication activity and leave further improvement to subsequent projects as *future work.*

In this thesis, the artifact is evaluated in terms of several factors. Extensive experiments are performed (Chapter 7) considering *running time* as a quantifiable measure of performance. These experiments aim to evaluate the ability of the GoPED artifact to work appropriately against a vast variety of event logs. The artifact is also evaluated in a

healthcare case study (Chapter 8) to compare the results with the defined objectives of the solution. A comparison with closely-related work is also provided (Chapter 9).

3.1.6 Activity 6: Communication

In this activity, the problem and its importance, the designed and developed artifact, its novelty and utility, and the rigour and effectiveness of its design are communicated to relevant audiences (e.g., researchers and practitioners). Knowledge of the disciplinary culture is required in the communication activity.

Different activities of this thesis, including the literature reviews [6, 3], the problem identification [7], the design [10], some components of the artifact [9], and the position of the artifact in the research field of combining goal modelling with business process modelling [11] were published in scholarly papers and presented in related conferences. This publicly available thesis also provides a comprehensive report of all the research activities.

3.2. Research Entry Point

Although the DSR process is structured in a nominally sequential order, it is not required that researchers always proceed sequentially from Activity 1 to Activity 6. In the real world, researchers may start with almost any of the aforementioned activities and move outward. Peffers *et al.* [5] identified four types of possible initiations that determine the appropriate entry point of research: *problem-centred initiation, objective-centred solution, design and development-centred initiation,* and *client/context-initiated*.

A *problem-centred initiation* is the basis of the nominal sequence of activities, where the research starts with Activity 1 (problem identification and motivation). The entry point of the research is problem-centred if the observation of a problem or suggested future work of previous projects results in the idea for the research. An *objective-centred solution* might be triggered when a research or industry need should be addressed by developing an artifact. In this case, the DSR process can start from Activity 2. The initiation is *design and development-centred* when an existing artifact has not yet been formally thought through as a solution for a specific problem domain where it will be used. Such an artifact might have already been used to solve a different problem, or it might have originated from another research domain. In this approach, the process might start from Activity 3. Finally, a

client/context-initiated solution may be based on the observation of an existing practical solution that worked. This approach, starting with Activity 4, might result in a solution if researchers work backwards to apply rigor to the process retroactively.

This thesis has a *problem-centred initiation* and started with problem identification and motivation. Our literature review revealed a research gap between the two contexts of goal modelling and process mining. As a consequence of such gap, the process models discovered in the process mining community do not consider systematically stakeholders' goal and their satisfaction levels. This results in process models that are not aligned with goals. These understandings were the basis of this thesis, and the problem was identified accordingly.

As the research was initiated by an identified problem, the nominal sequence of DSR activities is followed. However, it is not surprising that several iterations happened between the activities and even within one activity. In the next section, the iterations performed while developing this thesis will be explained.

3.3. Iterations Over the Activities

As Figure 20 shows, DSR is an iterative research method and does not dictate a strict sequence of activities. The researcher can start at almost any activity and move outward and also may iterate back over some activities that are already performed. Some of the iterations that happened in this thesis are explained in this section.

3.3.1 Iteration Over the Problem Identification

Given the capabilities of process mining revealed in the first literature review [6], at the very early stage of this research, we considered finding a method for mining the goals from the event logs, called *goal mining*. However, the second literature review [3] that considers existing work at the intersection of process mining and goal modelling convinced us that goal mining has already attracted some attention (e.g., *intention mining*) and the goal mining scheme that we considered had too many ambiguities. Also, the problem of unreliable rationality of discovered models was explicitly revealed. Accordingly, after the literature review, we iterated back over the problem identification.

In addition, after designing and developing the GoPED algorithms and method, we found that GoPED is essentially an event log filtering scheme. Therefore, we needed to iterate back to the literature review to find the state of filtering methods currently used in process mining activities2.8).

Finally, after designing and evaluating GoPED, we iterated back over the literature review and compared the capabilities of GoPED with the other related work that we had reviewed and highlighted the novelty and the new feature that GoPED provides (Section 9.1).

3.3.2 Iteration Over the Design and Development of GoPED

After the first demonstration of the main idea of the GoPED method in an illustrative example, we found that when some cases share a similar trace, but all of them are not selected with GoPED, theoretically, there is a possibility of bias that can decrease the quality of the discovered model. Therefore, we iterated over the design and ended up at adding GoPED's all-or-none rule, explained in Section 5.1.

Additionally, during the experiments (as an activity of demonstration and evaluation), we found some improvement ideas for the three GoPED algorithms. For example, the first version of GoPED only supported one comparison operator (less or equal to, $\leq$) for goal-related criteria for all three algorithms. Based on the needs of some experiments, the algorithms and their implementation were improved to support other operators as well (e.g., $\geq$, $\neq$, and $=$).

Also, after publishing the GoPED method paper [10], feedback from the process mining community helped us highlight the role of KPIs during the data preprocessing part of GoPED. Accordingly, the data preprocessing was elaborated, and two tools (explained in Section 4.2) were developed accordingly.

3.3.3 Iteration Over the Demonstrations and Evaluation of GoPED

For demonstration and evaluation, we first considered using real event logs of a hospital in Ontario. However, after obtaining a public event log from that hospital and considering it, we concluded that the event log did not have any goal-related attribute or even any potential

of deriving such attributes. In addition, our contact at that hospital moved to different organization soon thereafter. Therefore, we stopped working on that event log and looked for another log more appropriate for our research. We ended up selecting a publicly-available healthcare event log that had sufficient potential for deriving goal-related attributes.

Also, during the case study, we decided to extend the language for expressing goals supported in the GoPED tool while consulting with a physician and studying a medical guideline.

3.4. Chapter Summary

This chapter detailed how DSR was adapted to the context of this thesis, together with an explanation of the research method's activities, entry point, artefacts, and iterations.

The next chapter will consider the challenges that process mining, on one hand, and requirements engineering, on the other hand, can mitigate using synergetic effects achievable by combining goals and process mining.

Chapter 4. GoPED Fundamentals

This chapter explores how process mining can exploit goal modelling to guide process discovery (RQ) and identify opportunities for better aligning processes with goals. Our literature reviews in Chapter 2 showed us what came before and what has been considered by other researchers. One of the main objectives of the literature reviews was to find whether our research question (RQ) was partially or fully answered in the past. It was found that although process mining and goal modelling are growing research topics with considerable amounts of publications, there are only a few rare studies conducted at their intersection. This suggests that the techniques that exploit both data logs and goal models to produce meaningful processes can still be considered a gap to be filled.

In RQ, we are looking to propose a way to enrich process mining activities such that they exploit both event logs *and* measurable goals in order to generate models aligned with goals. Although some reviewed papers proposed some ways to mine the goals or intentions behind the actors' behaviour, this particular question has not been addressed yet.

4.1. Goal-oriented Process Enhancement and Discovery (GoPED)

Process discovery is the most prominent process mining technique. It takes an event log as input and produces a model without using any a priori information (see Figure 21). There exist many techniques to extract a process model from event logs, e.g., the α-algorithm [79]. Many organizations find it surprising that process discovery techniques can discover real processes from merely some executions recorded in data logs. Process discovery is often the starting point for other types of analyses [1].

Another type of existing process mining is *process enhancement*, sometimes called *process extension*. The main idea is to extend an existing process model using some information about the actual process recorded in some event logs. For example, one can derive the interaction patterns between individuals/roles by using the names of the roles or agents who have performed the activities recorded in the event log.

In the literature, some perspectives such as organizational, time, and resource perspectives have received some attention, but a *goal-oriented perspective* (including goals of actors, customers, and organizations) is not visibly considered.

In GoPED, the main idea is to exploit a wealth of information related to goals and their satisfaction levels that event logs may contain. In this method, in addition to the usual attributes of events, some cases' attributes recorded in the event log and correlated to some goals and indicators are also needed. Such related goals and indicators are assumed to be also contained by the corresponding goal model (e.g., specified using the Goal-oriented Requirement Language, part of the URN standard [40]).

In usual process enhancement activities, the information of new perspectives is added to the process model already discovered in the process discovery step. Therefore, adding new information does not have any impact on the discovery phase. In GoPED however, the goal-related information influences the process discovery step and, hence, the resulting process model.

While discovering a process model, the main concern of process mining algorithms is that the model should be "representative" of the behaviour realized in the event log [1]. Generally, when evaluating discovered models, there is a trade-off between four competing quality criteria. The model should allow generating the traces seen in the log (*Fitness)*, should not allow generating the traces irrelevant to log (*Precision)*, should not be too driven by examples in the log (*Generalization*) and, finally, should be the simplest model explaining the behaviour observed in the log (*Simplicity)*. There might not exist a model that satisfies all these criteria well. The intended usage of the process model might be the main basis for balancing these criteria.

Besides the fitness-related criteria, there are two other phenomena related to event logs that should be considered during process discovery: *noise* and *incompleteness*. Noise refers to the problem of having "*too much data*" that are describing rare behaviour, whereas incompleteness refers to the problem of having "*too little data*" to find the typical underlying process.

The steps of a process discovery algorithm, including the log preparation, and the algorithm's performance, are usually only subject to the criteria just explained. In GoPED,

there is a *new input* to process discovery. This input is a set of goals, including requirements, constraints, obligations, preferences, indicators, and stakeholder's expectations, derived from the goal model or documented requirements. The output of the method will still be a discovered process model. However, the discovered model, using goal-oriented information, not only meets the aforementioned quality criteria of process discovery, but also complies with the *intentions* acquired from the goal model.

In this approach, goals might be considered from different viewpoints, e.g., *customer* (case journey), *agent*, or *organization*. These viewpoints are enabled by the ability of languages to support modelling the goal of different actors, as in GRL.

As the new method enhances the usual data log by adding some goal-related attributes and then influences the *usual* steps of process discovery, it is called "goal-oriented process enhancement *and* discovery".

We hypothesize that a goal-oriented approach combined to process mining enables leveraging goals to improve process models and their realization. Process models that are discovered with respect to different goals are aligned with such goals and hence more likely to produce high levels of satisfaction.

In GoPED, goal satisfaction levels are derived from a model capturing goals, stakeholders, and their relationships. Note that the "enhancement" part of GoPED is about enhancing logs with goal information to produce higher-quality and simpler process models, and not about improving processes after their discovery.

As an example, a trace of activities in a healthcare process may take a very short time (i.e., it satisfies the goal "to decrease process time") but may end up with a wrong diagnosis (i.e., it violates the goal "to diagnose correctly"). Inversely, a trace may take a long time and impose an unaffordable cost but may end up with a correct diagnosis. GoPED takes advantage of goal models to manage such conflicting goals and to support trade-off analysis. With GoPED, good historical experiences will be found within the whole event log to be used as a basis for inferring good models, whereas bad experiences will be found to be avoided.

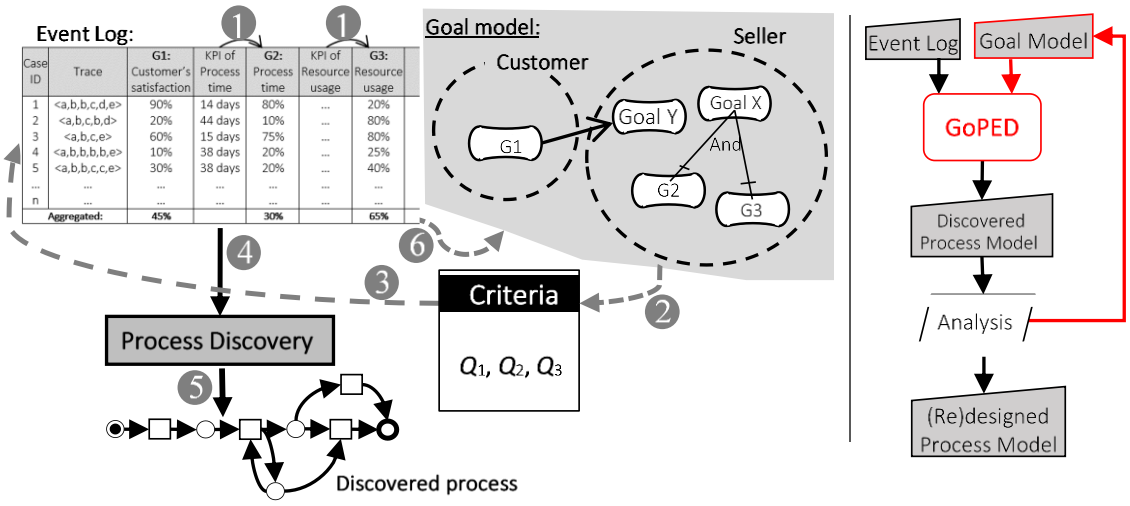


Figure 21. Overview of goal-oriented process enhancement and discovery (GoPED)

The *goodness* of traces and models is defined with regards to three categories of goal-related criteria: satisfaction of individual cases in terms of some goals (*case perspective*), overall satisfaction of some goals rather than individual cases (*goal perspective*), and a comprehensive satisfaction level for all goals over all cases (*organization perspective*). GoPED is expected to guide process discovery approaches towards specific goal-related properties of interest.

Figure 21 gives an overview of the method through an example that exploits the Goal-oriented Requirement Language (GRL) standard [2]. Let us assume that there are three leaf goals (G1, G2 and G3), with G1 contributing to Goal Y and Goal X being AND-decomposed by G2 and G3. Each goal may be fed by its Key Performance Indicators (KPIs), allowing to quantify its satisfaction level. Let us also assume that the event logs store the value of each goal-related KPI, e.g., "process time" associated to the goal G2 "to take a short process time". Such KPIs and how they contribute to goal satisfaction (e.g., by providing a function that converts a current, observable value to an abstract satisfaction value between 0/violated and 100/satisfied) are also defined in the goal model (see [55] for details). Based on this scheme, the satisfaction level of the leaf goals in Figure 21 are computed from corresponding KPIs (arrows ❶). We define the satisfaction level of Goal *i* as *Sat* (G_i). The satisfaction of actors (dashed circles in Figure 21) and of the whole models can be computed in a similar fashion [55].

After finding the current satisfaction of considered goals, GoPED defines some criteria related to the goals (arrow ❷). The main objective is to design a process model that fulfills one or many such criteria. The goal-related criteria are defined from three perspectives as follows:

- The resulting process model achieves (based on current evidence) a minimum satisfaction level for every single case in terms of one or multiple goals (*row* or *case perspective*). In Figure 21, a goal-related criterion could be, for example, "*Sat*(G_2) must be higher than 60" for all cases.
- The resulting process model achieves a threshold for the *aggregated* satisfaction level of one or multiple goals rather than the level for individual cases (overall *column* or *goal perspective*). In Figure 21, a goal-related criterion could be, for example, the aggregated satisfaction level of G2 (where the aggregation function is defined as the *average* here) must be higher than 70 (but is currently 45).
- The resulting process model achieves a threshold for the comprehensive satisfaction level of many goals over all cases (*table* or *organization perspective*, arrow ❻). This may be computed through the goal model (e.g., in GRL) or through a function derived from that model. For example, according to the structure of the model in Figure 21, the satisfaction of the stakeholder "Seller" is the average of Goal Y (computed from G1) and Goal X (the minimum of G1 and G2).

The basis of process mining is generally to use historical event logs and infer valuable insights. Following this general approach, GoPED selects a *subset* of the input traces that have already fulfilled the given criteria and uses them to find process models of interest (arrow ❹). For example, if the objective is to secure at least 50 as a satisfaction level for G2 for all the customers, the cases #1 and #3 will be selected because they have a satisfaction level over 50 for goal G2. After such a selection, a process model is mined through the selected traces using a process discovery algorithm (arrows ❹ and ❺). The discovered model does not represent all existing behaviours, but rather represents the desired behaviours towards the goals. Different model mined through different goal-related criteria can shed some light on different aspects and alternatives involved in the real or the discovered model. Such criteria are purposely defined by a domain expert in collaboration with a modeller. Moreover, an analyst can compare the model discovered from the whole log with the

model discovered by GoPED. Such a comparison can also reveal some valuable insights from potential discrepancies. Goal-oriented conformance checking approaches [7] [8] can also suggest some way of reconsidering the goal model with respect to misalignments between the process and goal perspectives, as shown in the right graph of Figure 21.

The process model resulting from GoPED is inferred from cases selected based on their goals. Therefore, irrelevant cases (that likely pursue goals different from the expected goals) are filtered out. The discovered model will be more likely well-structured as GoPED intentionally decreases the number of variations of traces and, in turn, decreases the chance of producing a spaghetti-like process model.

Another benefit of GoPED relates to the quality dimensions considered in usual process mining activities. In addition to the *fitness*, *precision*, *generalization*, and *simplicity* dimensions [174], GoPED brings into consideration a new *intention* dimension, formalized by the goal model.

In addition to the goal satisfaction, a process model that guarantees some specific *requirements* derived from goal models may be targeted as well. For example, a constraint may forbid performing activity *e* immediately after activity *b*, i.e., the trace ⟨…, *b*, *e*, …⟩ is not allowed. This kind of requirements might be represented in Linear Temporal Logic (LTL). LTL is a logic that, in addition to classical logical operators, also contains some temporal operators [1, 175]. van der Aalst *et al.* [176] took advantage of LTL to introduce an LTL-based language in the context of process mining. GoPED can enhance the process model through two potential paths (the two arcs leaving *Requirements* in Figure 21). The first path checks each trace, and the acceptable traces will constitute the input for process discovery. The second path, however, will prune the raw process model inferred form the raw log to satisfy one of the aforementioned criteria.

The process model resulting from GoPED is learned through some cases that are carefully selected based on their goals and satisfaction levels. Therefore, as irrelevant cases together with unsatisfied ones have been filtered out, the discovered model is more likely well-structured and, also, oriented to satisfy predefined goal-related criteria.

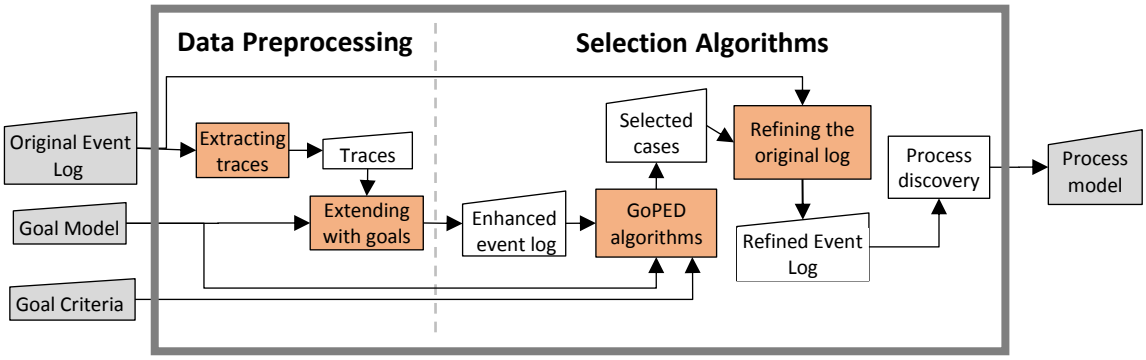


Figure 22. Overview of GoPED steps and the position of data preprocessing

4.2. Data Preprocessing for GoPED

According to the above conceptual overview, from an implementation perspective, the inputs, steps, and outputs of GoPED are shown in Figure 22. In this section, we introduce each element involved in the steps and describe their characteristics.

4.2.1 Inputs of Data Preprocessing

Original event log: Table 7 shows a fragment of an original event log related to the diagnosis of gestational diabetes (DGD) in healthcare. We assume that each row of the table shows an event and that each event needs to refer to a single process instance, called as *case*. We also assume that events are related to some activity. For example, in Table 7, events refer to activities such as admission, regular test, etc.

In process mining, "*case identifier*" and "*activity*" are mandatory columns. Furthermore, events related to a case need to be ordered. Hence, either a *timestamp* or a sequence number that shows the order of occurrence of the events (e.g., *event ID*) is also required. Without ordering the events, it is impossible to discover a process model. The first three columns of Table 7 show these three required attributes. An original event log can also contain additional event attributes that can be used for the analysis of discovered models (e.g., *resource*). Similarly, some attributes about the case (e.g., *age*) or about a case's trace (e.g., *total process time*) might be stored.

Table 7. An Example of an Original Event Log

Case	*Activity*	*Timestamp*	*Resource*	*Cost*	*...*
1	a: admission	12/03 09:35	Sarah	$150	
1	b: regular test	12/03 11:12	Tina	$100	
1	c: check the result	15/03 13:01	Hannah	$50	
1	g: send the result	16/03 08:35	Jane	$100	
2	a: admission	01/03 09:35	Rose	$150	
2	b: regular test	01/03 10:12	Tina	$100	
2	c: check the result	03/03 14:01	Hannah	$50	
2	g: send the result	06/03 08:15	Jane	$100	
3	a: admission	12/03 09:45	Rose	$150	
3	b: regular test	12/03 11:32	Tina	$100	
3	c: check the result	15/03 11:39	Hannah	$50	
3	d: ask advanced test	15/03 11:41	Hannah	$0	
3	e: advanced test	20/03 09:35	Linda	$400	
3	c: check the result	22/03 14:12	Hannah	$50	
3	g: send the result	23/03 08:17	Jane	$100	

As shown in Figure 22, the original event log with its three required columns is the main input of the preprocessing stage of GoPED.

Goal model: Typically, a goal model is a graph showing how different goals contribute to each other [42]. A goal model consists of stakeholders (or actors), their goals and contributing KPIs, AND/OR refinements, weighted contribution links, the importance level of goals to their stakeholders, and the stakeholders' importance in the whole model (see Figure 21). Such elements are the basis for evaluating the satisfaction level of goals through bottom-up analysis, as detailed in the goal-oriented modelling literature [55]. In a bottom-up analysis, the source nodes of the GRL model, often leaves in the graph, are provided initial satisfaction levels (or current values in the case of KPIs) that are propagated to higher-level intentional elements through the links connecting them. How to compute the satisfaction level of a goal based on its KPIs will be explained in Section 4.2.3.

Table 8. Event log enhanced with *n* goal-related attributes

Case	*Trace*	*Goal 1*	*Goal 2*	…	*Goal n*	*Overall*
c_1	t_1	$s_{1,1}$	$s_{1,2}$	…	$s_{1,n}$	$s_{1.Ove}$
c_2	…	$s_{2,1}$	$s_{2,2}$	…	$s_{2,n}$	$s_{2.Ove}$
…	…					…
c_m	t_m	$s_{m,1}$	$s_{m,2}$	…	$s_{m,n}$	$s_{m.Ove}$
Aggregated satisfaction		$s_{Agg.1}$	$s_{Agg.2}$	…	$s_{Agg.n}$	s_{Comp}

4.2.2 Output of Data Preprocessing

The objective of data preprocessing is to generate a table ready to feed to GoPED selection algorithms. Such a table, called *EnhancedLog*, is a version of the original event log that is first structured in a particular fashion and then extended with satisfaction levels of goals (including KPIs and actors). Table 8 shows the enhanced event log consisting of all cases (c_i) in the rows beside their traces (t_i). The satisfaction levels of all considered goals are stored in the next columns titled as "Goal *i*", which is the name of goal *i*. In such columns, $S_{i,j}$ shows the satisfaction level of goal *j* for case *i*. The last *Overall* column contains the satisfaction of the entire GRL model for each case. The bottom row in Table 8 aggregates (e.g., by computing the average, median, mode, etc.) the satisfaction levels of all cases for a given goal. We assume that *EnhancedLog* comprises m cases and n considered goals. The structure of the *EnhancedLog* table will be formally defined in Section 4.1.

4.2.3 Data Preprocessing

As Figure 22 shows, in order to produce enhanced logs, two transformations are executed: 1) extracting the traces, and 2) enhancing the traces with goal-related attributes.

Extracting traces

In this step, all events of a case are collected and ordered based on their execution time. Such a sequence is called a trace and is kept in a new table next to its case identifier. For

example, Table 7 contains 15 events that represent three cases, where each case includes events that denote the execution of healthcare activities. Each event has a time to be employed for ordering the activities. As shown in Table 8, cases c_1 and c_2 both refer to trace ⟨a, b, c, g⟩ while case c_3 refers to trace ⟨a, b, c, d, e, c, g⟩.

In the preprocessing step, '*extracting traces*' simply converts the original event log (Table 8) to a structured log (Table 9).

Extending with goals

As shown in Figure 22, this transformation takes goal-related attributes into consideration. In this step, for each goal considered in the goal model, one column is added to the table of traces. Then, KPIs associated to each goal are used to compute the satisfaction level of that goal for each case ($s_{i,j}$ in Table 8). Here, we elaborate on how to compute the satisfaction level of a goal based on its KPIs.

An indicator is a measure of the degree of fulfillment of stakeholders' goals. The indicators are needed to evaluate the satisfaction of goals. For example, the indicator "Process time" is needed to evaluate the goal "To increase process time". There is a set of parameters that should be evaluated against the *current value* of each indicator to find the satisfaction level of its corresponding goal: *target* value, *threshold* value, and *worst* value. The *target* value is the performance level expected from the process under consideration. The satisfaction level of each intentional element is a value in the range [0 to 100]. When a *current* value meets the *target*, the satisfaction level of the goal is 100. The *threshold* value, however, is used to separate acceptable values from unacceptable ones. When a process meets only the *threshold*, the satisfaction level is 50. The *worst* value specifies the worst condition from the stakeholder's perspective where the satisfaction level is zero. When we want to maximize an indicator (e.g., "To increase sales volume"), then *Target* ≥

Table 9. Traces extracted from the log shown in Table 8

Case	*Trace*
c_1	⟨a, b, c, g⟩
c_2	⟨a, b, c, g⟩
c_3	⟨a, b, c, d, e, c, g⟩

Threshold ≥ *Worst*. Likewise, when we aim to minimize an indicator (e.g., "To decrease cost"), then *Target* ≤ *Threshold* ≤ *Worst* [177].

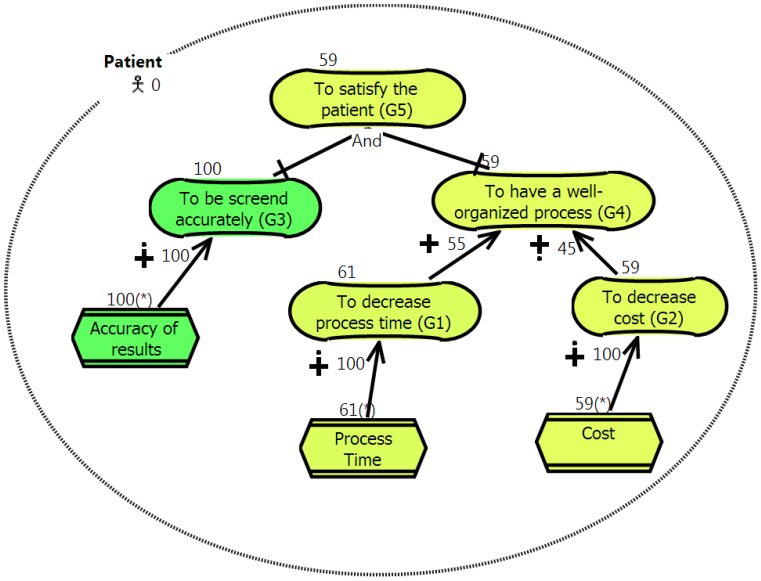


Figure 23. The goal model for the DGD process

Table 10. Current value of KPIs

Case	*Processing time*	*Cost*
c_1	4 days	$400
c_2	5 days	$400
c_3	11 days	$850

In this mapping scheme, from 0 (totally dissatisfied) to 100 (totally satisfied), the satisfaction level (*SL*) will be described by the equations below, coming from standard GRL:

For maximization:

$$SL = 100 \times \begin{cases} \frac{1}{2} + \frac{|Current\ v. - Threshold\ v.|}{2|Target\ v. - Threshold\ v.|}, & \text{if } Current\ v. \geq Threshold\ v. \\ \frac{|Current\ v. - Worst\ v.|}{2|Threshold\ v. - Worst\ v.|}, & \text{if } Current\ v. < Threshold\ v. \end{cases}$$

For minimization:

$$SL = 100 \times \begin{cases} \frac{1}{2} + \frac{|Current\ v. - Threshold\ v.|}{2|Target\ v. - Threshold\ v.|}, & \text{if } Current\ v. < Threshold\ v. \\ \frac{|Current\ v. - Worst\ v.|}{2|Threshold\ v. - Worst\ v.|}, & \text{if } Current\ v. \geq Threshold\ v. \end{cases}$$

Normally, the current value of a KPI is a number between the worst and the target values. In case a current value exceeds the target value, the satisfaction level will be truncated to 100 and if it is worse the worst value, the satisfaction level will be truncated to 0.

In GoPED, we assume that the set of *target, threshold,* and *worst* values for KPIs are included with the goal model and the *current* value is either extracted from the event log or provided in an additional table. In some cases, the satisfaction level of some goal might be directly available in some other tables.

In the DGD process, we can assume that the KPI "processing time" for each patient is the time between the first activity (admission) and the last activity (send the result). For example, the processing time of case#1 is 16/03h08:35 – 12/03h09:35 = 4 days. Also, KPI "cost" is the sum of the costs of all activities of a case. For example, the cost of case#1 will be \$150+\$100+\$50+\$100= \$400.

The current value of the above two KPIs were computed for the three cases, shown in Table 10. Let us assume that the goal "To decrease process time" is associated to the KPI "processing time" and the goal "To decrease cost" is related to the KPI "cost" (see Figure 23). Let us also assume that the *target* value for processing time is 4 days, the *threshold* is 13 days, and the *worst* value is 35 days. Using the above equations, the satisfaction level SL of the goal "To decrease process time" for case#2, whose current value in Table 10 is 5 days, is as follows:

$$\text{SL}= 100 \times (\frac{1}{2} + \frac{|5-13.|}{2|4-13.|}) = 94$$

The satisfaction levels of the two goals are computed in a similar way, shown in Table 11. The DGD process has also another goal whose satisfaction level is directly available: "To be screened accurately". Let us assume that the satisfaction level of three patients in terms of the goal *G3* (To be screened accurately) is 100 for all three patients (i.e., all three patients were screened truly). This data is also integrated in Table 11.

After computing the satisfaction level of all leaf goals based on their KPIs and integrating the satisfaction of the goals that are directly available, the cells of the goal columns in Table 8, $s_{i,j}$, are completely filled. In this state, to complete the *EnhancedLog*, the overall satisfaction level for cases, found in the rightmost column of *EnhancedLog* (Table 8), is yet to be computed. This column represents the satisfaction level of the whole goal

model for each case. As explained in Section 4.1 , that parameter is found through bottom-up analysis [55]. For example, a goal model consisting of the three goals discussed for the DGD process is shown in Figure 23 using the GRL language (the current numbers and colours in the figure reflect the evaluation for case#3, see Table 11).

In the goal model graph, the root shows the main goal and the sub-goals show the leaves. In such a goal model, the overall satisfaction level equals the satisfaction level of goal "To satisfy the patient". Based on this goal model and its AND/OR refinements and the weight of contributions, the satisfaction level of "To satisfy the patient" is computed as follows:

$$\text{Overall satisfaction level} = Sat\,(G_5)$$
$$= \text{Minimum}\,\big(Sat\,(G_3)\,,\ 0.55 \times Sat\,(G_1) + 0.45 \times Sat\,(G_2)\big)$$

This kind of evaluation is known as *bottom-up* or *forward propagation* in GRL. The jUCMNav tool is a graphical editor that can be employed for evaluating GRL models [178]. This tool also generates the arithmetic functions that enable the computation of each KPI, goal, actor, and overall model in different programming languages (including Java, C, and Python) as elaborated by Fan *et al.* [179].

Table 11. Satisfaction level of goals

Case	*Trace*	*G1*	*G2*	*G3*	*Overall*
c_1	⟨a, b, c, g⟩	100	100	100	100
c_2	⟨a, b, c, g⟩	94	100	100	97
c_3	⟨a, b, c, d, e, c, g	61	59	100	59

4.2.4 Tools

We developed two complementary tools to apply the scheme explained in this chapter on original event logs and resulting in an *EnhancedLog* ready to be fed to any of the three GoPED algorithms. Two more tools implementing the algorithms and reconstructing a filtered event log will be presented in later chapters. The tools are available for download at https://github.com/Mahdi-Ghasemi.

TraceMaker

TraceMaker is a tool implemented in Python to automate the "*Extracting traces*" step explained in Section 4.2.3. The input of this tool is a CSV table whose first three columns are *case identifier*, *activity name*, and *timestamp*, respectively (e.g., Table 7). The output is a CSV file of a table consisting of two columns: *Case identifier* and *Trace* (e.g., Table 9). As *TraceMaker* extracts the trace of each case from the original log, its output can be used not only for GoPED data preprocessing, but also for considering and analyzing traces before discovering the models. The output of *TraceMaker* is used to generate the input of the next tool that extends the traces by adding satisfaction levels.

The performance of the tool was evaluated using a real data log from the Children's Hospital of Eastern Ontario (CHEO). This log consists of 160,677 rows (events) representing 18 different activities and 11,842 patients (cases). The time taken by *TraceMaker* to extract the trace of all patients amounted to 0.59 seconds on a machine equipped with an Intel Core i7-2760QM CPU at 2.40 GHz, 8.00 GB RAM, and Windows 7. Doing this manually would typically take in the order of one hour, using different tools (such as R and Excel), and with higher risks of introducing mistakes.

EnhancedLogMaker

After generating the traces, *EnhancedLogMaker* can be used to compute the satisfaction level of goals for all cases as explained in Section 4.2.2. The input of this tool is a CSV file consisting of all cases beside their trace (generated by *TraceMaker*), the current values of all KPIs for all cases fill the remaining columns. The set of performance parameters for each indicator: *target* (value), *threshold* (value), and *worst* (value) are also stored in their corresponding column. The output will be a CSV file whose first two columns are *case identifiers* and *trace* and the next columns show the satisfaction level for all considered goals (e.g., Table 11). EnhancedLogMaker took 0.48 seconds to compute the satisfaction level of three goals for the 11,842 cases of CHEO's data on the same machine as above.

Although much testing remains to be done, these numbers on a realistic log suggest that these tools perform their transformations in pragmatic times and can be incorporated in larger automation processes. We have also tested two implementation alternatives, and

the one using Python's native *list()* construct was substantially faster than a previous version using *Pandas DataFrame*. Both versions are available for download.

4.3. Chapter Summary

This chapter introduced the fundamental concepts and steps underlying GoPED. It also defined the inputs and outputs of the various GoPED steps. This chapter focused on the data preprocessing step from the left part of Figure 22, and discussed its tool support. The next chapter will focus on GoPED's selection algorithms (right part of Figure 22), which produce, from the enhanced event logs that contain goal-related information, refined event logs that are readily processable by process mining tools.

Chapter 5. GoPED Algorithms

After the overview of GoPED method and its data preprocessing step presented in the previous chapter, in this chapter, the three GoPED algorithms are explained to enable selecting the best subset of cases where the goal-related criterion holds. For each type of criteria that suggest desired satisfaction levels from a (i) case perspective, (ii) goal perspective, and (iii) organization perspective, an algorithm is presented. To this end, first, the parameters and the concepts used throughout the algorithms are defined, then the algorithms are explained. Implementations of these algorithms are provided in Chapter 7.

5.1. Definitions

In process mining, the three attributes (columns) that must minimally exist in an event table are *case identifier*, *activity*, and *timestamp*. There might be some other event attributes stored in such a table that can be used for the analysis of discovered models (e.g., *resource*). Similarly, there might be some attributes about the case (e.g., age) or about a case's trace (e.g., total process time). In GoPED, we add some new case attributes related to goals, which are usually absent in process mining practice. Table 12 shows the architecture of event logs enhanced with goal-related attributes, used as input of GoPED. The notations that are used through this thesis are defined as follows:

Definition 1: Basic concepts (*activity*, *trace*, *case*, *event log*).

- A is the set of all experienced *activities* labelled a_i.
- A *trace* is a finite sequence of activities $t = \langle a_1, \cdots, a_k \rangle$, where $k \in \mathbb{N}^+$ is the trace length. T is the set of all observed traces.
- A *case* $c = \langle id,\ t \rangle$ has a unique case identifier $id \in \mathbb{N}^+$ and contains a trace $t \in T$.
- $trace(c) = t$ is a shorthand to indicate that the trace of the case c is $t \in T$.
- $L = \langle c_1, \cdots, c_m \rangle$ is an *event log* consisting of a finite sequence of cases of size *m.*
- C is the set of all possible cases (with traces) represented in the log L.

Definition 2: *EnhancedLog* structure.

To select the "best" subset of cases in a log through GoPED's algorithms, an event log enhanced with additional goal-related attributes is needed. The structure of such log, shown in Table 12, and the elements of that structure are defined as follows:

- *EnhancedLog* is the event log L enhanced with goal-related attributes. This log is a table of all cases $c \in C$ beside their traces $t \in T$. The satisfaction levels of all considered goals (including KPIs and actors) are stored in the next columns.
- $\mathbb{G}$ is the set of all considered goals, i.e., $\mathbb{G} = \{G_1, G_2, \cdots, G_n\}$. We assume that *EnhancedLog* consists of m cases and n considered goals.
- $s_{i,j}$ in *EnhancedLog* shows the level of satisfaction of *case*$_i$ in terms of *Goal*$_j$.
- $s_{i.Ove}$, found in the last column of the table *EnhancedLog*, is the overall satisfaction level for *case*$_i$. This represents the satisfaction level of the whole goal model. The satisfaction level of the goal model is evaluated through bottom-up analysis as elaborated in the goal-oriented modelling literature [55]. This evaluation is based on AND/OR refinements, contribution links, the importance level of a goal to its actor, and the actor importance in the whole model.
- Function g is derived from the goal model to compute the overall satisfaction level based on satisfaction levels of all sub-goals in $\mathbb{G}$ [179]. Therefore, as $s_{i,j}$ is the satisfaction level of *Goal*$_j$ for *case*$_i$, we have $s_{i.Ove} = g(s_{i.1}, s_{i.2}, \cdots, s_{i.n})$.
- $s_{Agg.i}$, in the last row of *EnhancedLog*, show the aggregated satisfaction level of each goal based on the satisfaction level of all cases in terms of that goal. $s_{Agg.i}$ is a function (e.g., average, median, etc.) of satisfaction levels of all m cases for *Goal*$_j$.

Table 12. Event log enhanced with *n* goal-related attributes (*EnhancedLog*)

Case	*Trace*	*Goal 1*	*Goal 2*	…	*Goal n*	*Overall*
c_1	t_1	$s_{1,1}$	$s_{1,2}$	…	$s_{1,n}$	$s_{1.Ove}$
c_2	…	$s_{2,1}$	$s_{2,2}$	…	$s_{2,n}$	$s_{2.Ove}$
…	…					…
c_m	t_m	$s_{m,1}$	$s_{m,2}$	…	$s_{m,n}$	$s_{m.Ove}$
Aggregated satisfaction:		$s_{Agg.1}$	$s_{Agg.2}$	…	$s_{Agg.n}$	s_{Comp}

- Function f_j is the aggregation function of $Goal_j$. For each goal $G_j \in \mathbb{G}$, we have $s_{Agg.j} = f_j(s_{1.j}, s_{2.j}, \cdots, s_{m.j})$. F is a tuple of functions $(f_1, f_2, ..., f_n)$ that keeps aggregation functions of all goals.
- s_{Comp} is the comprehensive satisfaction level that the process has yielded. This factor can be defined either by composing the aggregated satisfactions (last row) or by aggregating the overall satisfaction levels (last column) using some function.

Definition 3. GoPED offers three types of goal-related criteria, introduced in Section 4.1:

- $\boldsymbol{Q_{case}}$ is the logical conjunction (AND) of a set of criteria q_j. Each q_j is a tuple composed of one goal $G_j \in \mathbb{G}$, one comparison operator $\Diamond_j \in \{<, \leq, >, \geq, =, \neq\}$, and a threshold sl_j between zero and 100, for the satisfaction level of that goal. The Q_{case} can be shown as:

  $$Q_{case} = \{q_j = (G_j, \Diamond_j, \bar{sl}_j) \mid G_j \in \mathbb{G},\ \Diamond_j \in \{<, \leq, >, \geq, =, \neq\}, 0 \leq \bar{sl}_j \leq 100\}.$$

  A confidence level, $0 \leq conf \leq 1$, together with Q_{case}, constitute the whole criteria. Such criteria represent that (with a confidence *conf*) the satisfaction level of every single case in terms of the considered goals G_j will hold the condition that $\Diamond_j$ and $\bar{sl}_j$ define. For example, when $\mathbb{G}$ = {G_1, G_2, G_3} and Q_{case}= {(G_2, ≥, 70), (G_3, ≤, 80)} and *conf* = 0.8, GoPED is looking for a process model that will yield minimum satisfaction levels of 70 for G2 and maximum satisfaction levels of 80 for G3, for at least 80% of the cases (i.e., confidence of 0.8). It is noteworthy that all goals in $\mathbb{G}$ are not necessarily considered by Q_{case}.
- $\boldsymbol{Q_{goal}}$ refers to the second type of goal-related criteria. Q_{goal} is the logical conjunction (AND) of a set of criteria q_j composed of one goal $G_j \in \mathbb{G}$, one comparison operator $\Diamond_j \in \{\leq, =, \geq\}$, and a satisfaction level for that goal. Therefore,

  $$Q_{goal} = \{q_j = (G_j, \Diamond_j, \bar{sl}_j) \mid G_j \in \mathbb{G},\ \Diamond_j \in \{\leq, =, \geq\}, 0 \leq \bar{sl}_j \leq 100\}.$$

  Q_{goal} is looking for a process model that can deliver an *aggregated* satisfaction level for the considered $G_j \in \mathbb{G}$ of at least/at most/equal-to $\bar{sl}_j$. For example, when $\mathbb{G}$ = {G_1, G_2, G_3} and Q_{goal}= {(G_2, ≥, 90), (G_3, ≥, 75)}, Q_{goal} is looking for a process model that will yield minimum *aggregated* satisfaction levels of 90 for G2 and 75 for G3. Again, all goals in $\mathbb{G}$ are not necessarily considered by Q_{goal}.

- $\boldsymbol{Q}_{\mathbf{Comp}}$ consists of one value between 0 and 100 called $\overline{sl}_{Comp}$ and one comparison operator $\Diamond \in \{\leq, =, \geq\}$. This criterion looks for a process model that can yield a *comprehensive* satisfaction of at least/at most/equal-to $\overline{sl}_{Comp}$. Therefore,

$$Q_{Comp} = \{ \Diamond \in \{\leq, =, \geq\}, 0 \leq \overline{sl}_{Comp} \leq 100\}.$$

  For example, when $Q_{Comp} = \{\geq, 90\}$, the *comprehensive* satisfaction level should be at least 90.

Definition 4. *SelectedCases* $\subseteq C$ is the main output of GoPED algorithms and the set of selected cases that satisfy one of the aforementioned criteria.

Definition 5. *All-or-none* rule.

One very important feature of our search approach is that we look over the *cases* $\in C$ rather than the *traces* $\in T$. This is because many cases might share the same trace but have different levels of satisfaction for the goals. Moreover, the frequency of each trace contains very important knowledge about real-world behaviours. Therefore, we need to end up with a subset that consists of variations of traces together with their frequencies.

One consequence of searching at the level of cases is that there might be some cases with trace t_k that are eligible to be selected and, simultaneously, some cases with the same trace that are not eligible. Although including the former cases and excluding the latter ones appears to be a simple solution, it would not be correct. The reason is that a discovered model either allows a trace (and all its cases) or avoids it. Recall that process discovery algorithms work based on the "*direct succession*" between activities in traces. Some algorithms (e.g., α-algorithm) have a strong learning bias as they only focus on direct succession (regardless of their frequency), and some algorithms (e.g., Fuzzy Miner) work with direct succession beside their frequency.

Let us assume that (based on goal criteria) there are 10 eligible cases that are sharing the trace $\langle a, b, c, g\rangle$, and that there are 500 ineligible cases that are sharing the same trace $\langle a, b, c, g\rangle$. If a filtering algorithm selects the 10 eligible cases and these selected cases are fed to the α-algorithm, the resulting model will allow the trace $\langle a, b, c, g\rangle$, while about 98% of cases (500 out of 510) that experienced this trace are not aligned with the considered

goals. Therefore, the selection of only one case of a trace is enough to discover a model that allow such a trace.

A frequency-based discovery algorithm may handle that specific example, but there are some situations that would not be handled by such algorithms. For example, in the event log L= [$\langle a, b, c, g\rangle^{900}$, $\langle a, b, c, d, g\rangle^{200}$], suppose that the number of satisfied cases with the first trace is 250 (i.e., 650 cases are not satisfied), and 180 cases of the second trace are satisfied and eligible to be selected. If one simply selects only the satisfied cases and make C*= [$\langle a, b, c, g\rangle^{250}$, $\langle a, b, c, d, g\rangle^{180}$], the frequency of the first trace is high enough to affect the resulting model. Such a model will allow the traces $\langle a, b, c, g\rangle$ while only 28% (250/900) of cases of $\langle a, b, c, g\rangle$ are satisfied. Such a ratio for trace $\langle a, b, c, d, g\rangle$ is 90% (180/200). Accordingly, we may conclude that skipping activity *d* after *c* has a negative impact on our goal. Hence, the simply selected C*= [$\langle a, b, c, g\rangle^{250}$, $\langle a, b, c, d, g\rangle^{180}$] would result in a model that allows this skip and, consequently, the goal-related criteria will not be secured.

Coping with such a challenge, we respect an "*all-or-none*" rule, i.e., the *Selected-Cases* should have either all cases of a same trace or none of them. This rule is basically respected with the current filtering scheme, explained in Section 2.8.2, as such filtering methods are applied on traces to decide whether a trace must be kept or removed. If a trace is not selected, all of its instances (cases) in the log will be removed, whereas if it is selected all of its instance will be kept.

The decision about whether a trace and all of its cases can be selected can also be relaxed in some situations using a confidence level *conf*, which represents the minimal proportion of cases in a trace that must satisfy the search criterion for all the cases in that trace to be kept. When conf equals 100%, then all cases must satisfy the criterion.

Based on the above explanation, we define the three algorithms for selecting the best subset of cases regarding the three types of goal criteria and the *all-or-none* rule.

5.2. GoPED Algorithms

As the goal-related criteria are based on three different viewpoints of *EnhancedLog*, three different algorithms for trace selection are required. The main idea in all three algorithms is to select the largest subset of cases that satisfy the selected criterion.

Searching for the *largest* subset of cases is needed because if one simply selects very few cases that meet the desired criteria, the discovered model will be based on an event log suffering from potential *incompleteness* problems. When the event log consists of too few events, the discovered model is less realistic and risks becoming *overfitted*.

5.2.1 Algorithm 1: Guaranteeing One or Multiple Goals for All Cases

This first goal-related criterion is looking for a model that guarantees (with a given confidence level) a predefined satisfaction level for one or multiple goals for all cases. This criterion considers every single case in a row viewpoint, therefore each case will be assessed against all goals considered in the criterion. There might be cases with trace t_k that meets all $q_i \in Q_{\text{case}}$ and some cases with the same trace that do not. Algorithm 1 checks all cases of one trace against Q_{case} (line 11). If the proportion of complying cases is not inferior to the given confidence level *conf*, all the cases with that trace will be selected, otherwise all of them will be filtered out (lines 16-17).

For example, assume *conf* is 0.8 and the event log has 100 cases with trace $\langle a, b, c, g \rangle$, including 83 cases that meet Q_{case} and 17 cases that do not. As 83% of these cases comply with the criterion, which is above the confidence level of 80%, all 100 cases are selected. Algorithm 1 first sorts all cases according to their trace (line 1). Searching within all cases of a trace and checking them against the criteria is hence efficient (lines 9-15).

Algorithm 1 *Selecting a subset of an event log to infer a process model that guarantees a minimum satisfaction level for one or multiple goals in each selected case*

Input: *EnhancedLog*: Enhanced structured event log ▷ explained in Definition 2
Q_{case}: A set of criteria; ▷ explained in Definition 3
conf : A confidence level ▷ explained in Definition 3

Output: *SelectedCases* ▷ a subset of cases selected according to the criteria and the all-or-none rule

```
1   sort_by_trace(EnhancedLog) ▷ sort the cases based on their traces
2   trace(case_0) ← ⟨⟩ ▷ ⟨⟩ is an empty trace, which cannot happen in reality
3   trace(cases_NumberOfCases+1) ← ⟨⟩ ▷ also flag the end of the log
4   SelectedCases ← ∅
5   index ← 1
6   while index ≤ NumberOfCases ▷ NumberOfCases is m in Table 12
7   |  SameTraceCases ← ∅ ▷ a set of cases whose traces are the same
8   |  NumberOfSatisfiedCasesOfTrace ← 0 ▷ counts the satisfied cases of a trace
9   |  do
10  |  |  SameTraceCases ← SameTraceCases ∪ {case_index}
11  |  |  if case_index meets all criteria of Q_case then
12  |  |  |  NumberOfSatisfiedCasesOfTrace + +
13  |  |  end if
14  |  |  index + +
15  |  while trace(case_index) = trace(case_index−1)
16  |  if NumberOfSatisfiedCasesOfTrace / size(SameTraceCases) ≥ conf then
17  |  |  SelectedCases = SelectedCases ∪ SameTraceCases
18  |  end if
19  end while
20  return SelectedCases ▷ the resulting subset of cases
```

5.2.2 Algorithm 2: Guaranteeing the Aggregated Satisfaction Levels of Goals

Here, in a column perspective, the focus is on the *aggregated* satisfaction level of one or multiple goals. Logically, the largest subset that simultaneously meets all criteria is the intersection of the largest subsets that separately meet all criteria. Therefore, one can focus on all considered goals individually and find the largest subsets for each G_j regarding $\overline{sl}_j$, and then use their intersection as *SelectedCases*. However, finding the largest subset is not trivial because the largest subset that satisfies the criterion of one goal might not be unique, and different subsets with similar (and largest) sizes may satisfy the condition. This might be the case even when the aggregation function is simple (e.g., *average*). In this situation, a subset that makes the largest intersection of all subsets (generated by all considered goals) should be selected. When this situation happens for several goals, we must deal with the challenge of selecting one combination that finally makes the largest set.

Algorithm 2 *Selecting a subset of an event log to infer a process model that guarantees the overall satisfaction level(s) of one or multiple goals*

Input: *EnhancedLog*: An enhanced structured event log ▷ explained in Definition 2
Q_{goal}: A set of criteria (some goals & thresholds for their satisfaction level) ▷ Definition 3
g: A function computing the satisfaction of the whole goal model ▷ Definition 3

Output: *SelectedCases* ▷ a subset of all cases selected regarding the criteria and the all-or-none rule, *SelectedCases* ⊆ *C*, Definition 4

1 *SelectedCases* ← ∅
2 **Solve the binary optimization below:** (x_i is a flag for either selecting case$_i$ or not)

$Max\ z = \sum_{i=1}^{m} x_i$ ▷ this is to find the largest subset
s.t.
$\forall\ r, t\ \ 1 \le r < t \le m$: if $trace(c_r) = trace(c_t)$ $x_r = x_t$ ▷ all-or-none rule
$\forall\ j$ where $G_j \in \mathcal{G}$: $\frac{\sum_{i=1}^{m} x_i . s_{i,j}}{\sum_{i=1}^{m} x_i} \ge \overline{sl}_j$ ▷ $|Q_{goal}|$ constraints
$x_i = 0, 1$ ▷ if $x_i = 1$, case i should be selected

3 **end of binary optimization**
4 **for** i = 1 to NumberOfCases **do** ▷ NumberOfCases is m in Table 12
5 **if** x_i = 1 **then**
6 *SelectedCases* ← *SelectedCases* ∪ {c_i}
7 **end**
8 **end**
9 **return** *SelectedCases* ▷ the resulting subset of cases that meets the criteria

Addressing such a difficulty, Algorithm 2 generates a *binary optimization* whose number of variables (x_i) equals the number of cases (m). The binary variable x_i is a flag variable associated to case c_i. If $x_i = 1$, then the case c_i will be selected and, if not, it will be excluded. As we are looking for the largest subset, $\sum x_i$ should be as high as possible. There are two categories of constraints for the optimization problem. The first aims to preserve the *all-or-none* rule, i.e., the selected subset should include either all cases of a same trace or none of them. The second category of constraints takes care of the threshold for the aggregated satisfaction level of each goal, i.e., $\overline{sl}_j$. This category of constraints is based on f_j, i.e., the aggregation function. In Algorithm 2, we assumed that all f_j are the *average* function (but others could be defined).

5.2.3 Algorithm 3: Guaranteeing Comprehensive Satisfaction Levels

The two previous types of criteria considered the goals from a row perspective and a column perspective, respectively. The third type of criteria, however, considers the goals from a *table* perspective.

Here, the overall satisfaction level of all columns is aggregated and represented by one number as a *comprehensive satisfaction level*. Finding the largest subset of cases that

Algorithm 3 *Selecting the largest subset of an event log to infer a process model that guarantees a comprehensive satisfaction level*

Input: *EnhancedLog*: An enhanced structured event log; ▷ as explained by Definition 2
Q_{Comp} : Consists of a value between 0 and 100 called $\overline{sl}_{Comp}$ and a comparison operator $\Diamond \in \{\leq, =, \geq\}$.
F: a tuple of functions $(f_1, f_2, ..., f_n) \mid s_{Agg.j} = f_j(s_{1,j}, s_{2,j}, \ldots, s_{m,j}), j = 1,\ldots,n$
g: a function derived from the goal model. $s_{i.\text{ove}} = g(s_{i,1}, s_{i,2}, \ldots, s_{i,n})$
Output: *SelectedCases* ▷ a subset of all cases selected regarding the criteria and the all-or-none rule, $SelectedCases \subseteq C$

1 *SelectedCases* ← ∅
2 **if** $Q_{Comp} = f_{n+1}(s_{1.\text{ove}}, s_{2.\text{ove}}, \ldots, s_{m.\text{ove}})$ **then** ▷ special case similar to Algo. 2
3 use Algorithm 2 and exit.
4 **else**
5 **Solve the binary optimization below:**

$$Max\ z = \sum_{i=1}^{m} x_i \quad \triangleright \text{ this is to find the largest subset.}$$

s.t.

$$\forall\, r, t\ \ 1 \leq r < t \leq m: \text{ if } trace(c_r) = trace(c_t) \quad x_r = x_t \ \triangleright \text{ all-or-none rule}$$

$$g\left(\frac{\sum_{i=1}^{m} x_i.s_{i1}}{\sum_{i=1}^{m} x_i}, \frac{\sum_{i=1}^{m} x_i.s_{i2}}{\sum_{i=1}^{m} x_i}, \ldots, \frac{\sum_{i=1}^{m} x_i.s_{in}}{\sum_{i=1}^{m} x_i}\right) \Diamond\ Q_{Comp}$$

$x_i = 0, 1$

6 **end**
7 **for** $i = 1$ to NumberOfCases **do** ▷ NumberOfCases is m in Table 12
8 **if** $x_i = 1$ **then**
9 *SelectedCases* ← *SelectedCases* ∪ $\{c_i\}$
10 **end**
11 **end**
12 **return** *SelectedCases* ▷ the resulting subset of cases that meets the criterion

guarantees a minimum threshold for comprehensive satisfaction level ($\overline{sl}_{comp}$) is, also, not trivial. This is because adding a trace to the subset might increase the aggregated level of one goal and, at the same time, decrease the level of another goal.

As explained in Definition 3, the *comprehensive* level might be calculated in two different ways. It can be the overall satisfaction level of the aggregated levels of all goals, or the aggregated level of the overall satisfaction level of each case. The latter one will work only with the column of the overall satisfaction level (the right column of Table 12), therefore, the problem will be solved in a way similar to that of the second type of criteria (Algorithm 2). Accordingly, Algorithm 3 first checks the definition of *comprehensive* (lines 2-3). If it is not like Algorithm 2, then Algorithm 3 generates a new binary optimization problem. Here, the first category of constrains aims to preserve the *all-or-none* rule,

whereas the last constraint makes sure that the comprehensive level of the selected subset is not less than the given threshold $\overline{sl}_{comp}$.

5.3. Chapter Summary

This chapter defined three alternative GoPED algorithms for the selection of cases and traces in an enhanced event log that contains goal-related information (right part of Figure 22). These algorithms enable filtering out non-goal-compliant cases based on either 1) a case perspective (rows), 2) a goal perspective (columns, with aggregation functions), or 3) a comprehensive perspective (tables, also with aggregation functions). The all-or-none rule is enforced. Algorithms 2 and 3 require the use of binary optimization to properly support that rule. The next chapter will illustrate their application on a healthcare-related example.

Chapter 6. Illustrative Example

In this chapter, the *diagnosis of gestational diabetes* (DGD) process, provided in the healthcare system of Ontario and considered in Section 2.1.7, is used to illustrate the Go-PED method. To this end, the three algorithms corresponding to different types of goal-related criteria discussed in Chapter 5 are taken into consideration. Here, the objective is to discover a process model fulfilling the pre-defined criteria. Please note that the log is realistic but *not real* (e.g., without private data) and is used only to study the GoPED method and its algorithms.

6.1. Event Log of an Illustrative DGD Process

The event log of 10 patients who have used the DGD process is shown in Table 13. We use short names to encode the activities: *a* = admission, *b* = regular test, *c* = check the result, *d* = request for advanced test, *e* = advanced test, *f* = request for repetition, and *g* = send the result. According to Table 13 and the definitions provided in Section 5.1, the event log (L) includes five different variants of traces:

$$L = [\langle a, b, c, g\rangle^3, \langle a, b, c, d, e, c, g\rangle^3, \langle a, b, c, f, b, c, g\rangle^1,$$
$$\langle a, b, c, f, b, c, d, e, c, g\rangle^2, \langle a, b, c, d, b, c, d, e, c, d, e, c, g\rangle^1]$$

Table 13. Event log of 10 patients of DGD with the current values of KPIs

Case	Trace	*Process Time* (days)	*Cost* ($)	*Patient rating* (1-10)	Accuracy of results
Patient_1	$\langle a, b, c, g\rangle$	4	400	9	1
Patient_2	$\langle a, b, c, g\rangle$	5	400	9	1
Patient_3	$\langle a, b, c, g\rangle$	5	400	9	0
Patient_4	$\langle a, b, c, d, e, c, g\rangle$	11	850	8	1
Patient_5	$\langle a, b, c, d, e, c, g\rangle$	9	850	7	1
Patient_6	$\langle a, b, c, d, e, c, g\rangle$	10	850	8	1
Patient_7	$\langle a, b, c, f, b, c, g\rangle$	8	600	7	1
Patient_8	$\langle a, b, c, f, b, c, d, e, c, g\rangle$	17	1100	6	1
Patient_9	$\langle a, b, c, f, b, c, d, e, c, g\rangle$	16	1100	5	1
Patient_10	$\langle a, b, c, d, b, c, d, e, c, g\rangle$	31	1150	4	1

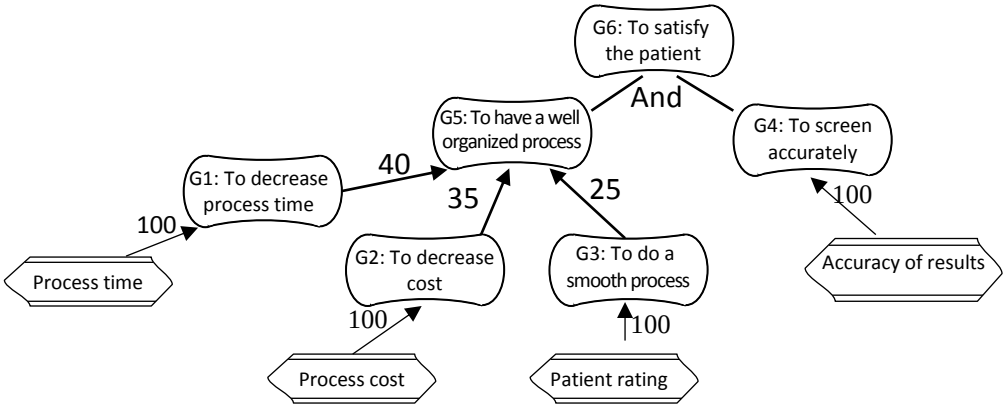


Figure 24. Goal model related to the DGD process

.

6.2. Goal Model Associated with the DGD Process

Figure 24 shows the goals model related to the DGD process using the Goal-oriented Requirement language. In the graph, the root is the main goal, and the KPIs are the leaves. As the goal model shows, the overall satisfaction level of a patient is evaluated by goal G_6 "To satisfy the patient". The goal model has an AND-type decomposition link between goals "To screen accurately" and "To have a well-organized process" to the main goal of "To satisfy the patient", i.e., $(G_4, G_5) \xrightarrow{AND} G_6$. As discussed in Section 2.2.1, the satisfaction level of a goal with an AND-type decomposition link is the minimum value of the satisfaction level of its source elements [55]. Hence, we have Sat(G_6) = *min*{Sat(G_4), Sat(G_5)}. On the other hand, three goals G_1, G_2, and G_3 are linked to G_5 through contribution links with contribution levels of 40, 35, and 25, respectively. In GRL, the total quantitative contribution is the normalized sum of the products of the satisfaction level of each source element by its contribution level to the element, truncated to range from 0 to 100 [55]. Therefore,

$$\text{Sat}(G_5) = min\{100, max\{0, 0.4 \times \text{Sat}(G_1) + 0.35 \times \text{Sat}(G_2) + 0.25 \times \text{Sat}(G_3)\}\}.$$

Overall, the satisfaction level of the main goal is computed through this function:

$$\text{Sat}(G_6) = min\{\text{Sat}(G_4), min\{100, max\{0, 0.4\times\text{Sat}(G_1)+0.35\times\text{Sat}(G_2)+0.25\times\text{Sat}(G_3)\}\}\}$$

Table 14. Definitions of the KPIs in the DGD process

Indicator	*Corresponding goal*	*Worst v.*	*Threshold v.*	*Target v.*
Process time (min)	To decrease process time	35	13	4
Cost ($)	To decrease cost	1200	950	400
Patient rating	To have a smooth process	1	6	10
Accuracy of results	To screen accurately	0	-	1

This kind of evaluation is known as *forward propagation* in GRL. The jUCMNav tool is an Eclipse-based graphical editor that can be used for evaluating GRL models [178].

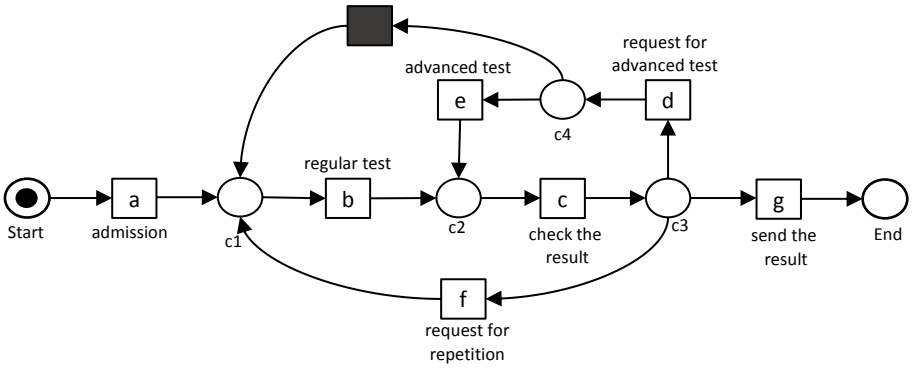
As Figure 24 shows, the four leaf goals, are connected to their corresponding KPIs. As each goal is connected to only one KPI, that KPI alone will define the satisfaction level of the goal (i.e., the weight is 100).

6.3. Converting KPIs to Satisfaction Levels of Goals

As discussed in Section 4.2.3, in GoPED, an analyst should find the satisfaction level of each leaf goal from its contributing KPIs. Table 14 shows the set of *target*, *threshold*, and *worst* values for the goals of the DGD process. According to the values shown in Table 14, the current values (Table 13) and the equations shown in Section 4.2.3, the satisfaction level of each goal was computed and is shown in Table 15. As the *current*, *target* and *worst*

Table 15. EnhancedLog: event log and satisfaction level of goals for the DGD process

Case	*Trace*	G_1: *To decrease process time*	G_2: *To decrease cost*	G_3: *To do a smooth process*	G_4: *To screen accurately*	***Overall*** (To satisfy the patient)
Patient#1	⟨*a, b, c, g*⟩	100	100	88	100	97
Patient#2	⟨*a, b, c, g*⟩	94	100	88	100	95
Patient#3	⟨*a, b, c, g*⟩	94	100	88	0	0
Patient#4	⟨*a, b, c, d, e, c, g*⟩	61	59	75	100	64
Patient#5	⟨*a, b, c, d, e, c, g*⟩	72	59	63	100	65
Patient#6	⟨*a, b, c, d, e, c, g*⟩	67	59	75	100	66
Patient#7	⟨*a, b, c, f, b, c, g*⟩	78	82	63	100	76
Patient#8	⟨*a, b, c, f, b, c, d, e, c, g*⟩	41	20	50	100	36
Patient#9	⟨*a, b, c, f, b, c, d, e, c, g*⟩	43	20	40	100	34
Patient#10	⟨*a, b, c, d, b, c, d, e, c, g*⟩	9	10	30	100	15
Aggregated satisfaction:		65.9	60.9	66	90	**64.1**



Model 1. Process model discovered from the original event log by the α-algorithm

values for the indicator "Accuracy of results" is binary, there is no threshold value defined for this indicator. In this case, satisfaction level takes a binary value (i.e., either 0 or 100).

An advanced version of the *α*-algorithm [180] generates Model 1 using the whole event log. The main story of this DGD process is as follows: after admission of a patient, a regular blood test is done. Then, based on the result of the test the patient may need to do an advanced test, the patient may need to repeat the regular test, or the result will be sent to the related department, and then the process ends. A *silent* transition is shown in Model 1 in black colour. That is a particular transition not observable in the event log, but needed to make a *sound* Petri-net [174]. Considering the traces, the source of this need is that for Patient#10, after the activity *d*, request for advanced test, the activity *b*, regular test, has executed, while the activity *e*, advanced test, was supposed to execute. Model 1 will be used as a basis for considering the resulting models from GoPED according to three types of goal-related criteria.

6.4. Example Models Resulting from GoPED

6.4.1 Guaranteeing Satisfaction of One or Multiple Goals for All Cases

The first type of goal-related criteria is looking for a model that guarantees a predefined satisfaction level for all cases in terms of one or multiple goals, with a given confidence level. For example, the condition is as follows:

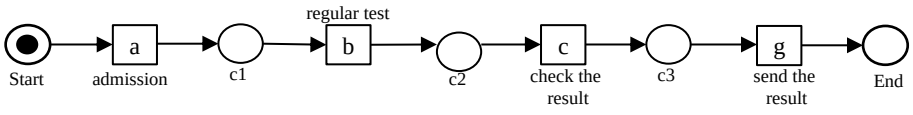
- *Case perspective:* generate a model that guarantees (with a minimum confidence of 90%) that the satisfaction level for all patients in terms of goal G1 "To decrease process time" will be at least 75 and that in terms of goal G3 "To do a smooth process" will be at least 80.

In this case, we have Q_{case}= {(G_1, 75), (G_3, 80)} and *conf* = 0.9. Using Algorithm 1, only all the cases of trace ⟨*a, b, c, g*⟩ are returned, i.e., Patients #1, #2 and #3. All cases of this trace meet $Q_{case,}$ so the fraction of eligible cases of this trace is 100%, which is more than the required 90% confidence level. Such a parameter for the four remaining traces is zero, which is less than 90% by far. Therefore, we have *SelectedCases*= {Patient#1, Patient#2, Patient#3}, resulting in the log {$\langle a, b, c, g\rangle^3$}. The *α*-algorithm [180] produces Model 2 from this log. This is the successful process to be encouraged in the organization to meet these goals.

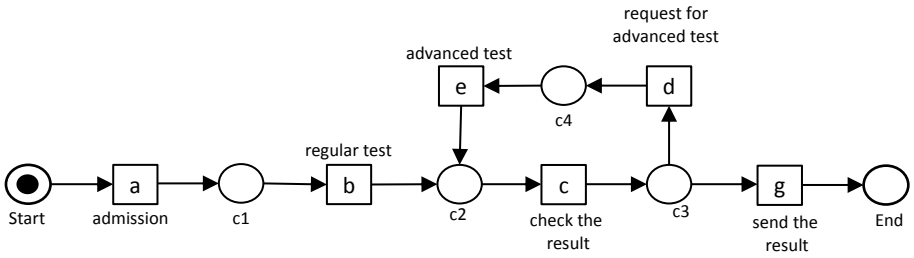
6.4.2 Guaranteeing the Aggregated Satisfaction Levels of Goals

Here, from a column perspective, the focus is on the aggregated satisfaction level of one or multiple goals rather than on the satisfaction of every single case.

- *Goal perspective:* Generate a model that results in an *aggregated* satisfaction levels of the goal "To decrease time process" higher than 80 and of the goal "To do a smooth process" higher than 78, simultaneously.



Model 2. To satisfy goal criteria of Q_{case}



Model 3. To satisfy goal criteria of Q_{goal}

In this case, we have Q_{goal} = {(G_1, 80), (G_3, 78)}. Here the functions *f* showing how to calculate the aggregation of each column is required. Let us assume that for all goals in the DGD process, the function is the *average*. Therefore, the optimization problem of Algorithm 2 can be formalized as follows:

$$Max\ z = \sum_{i=1}^{10} x_i$$

s.t.

$x_1 = x_2 = x_3$, $x_4 = x_5 = x_6$, $x_8 = x_9$ ← (*all-or-none* rule)

$$\frac{100x_1+94x_2+94x_3+61x_4+72x_5+67x_6+78x_7+41x_8+43x_9+9x_{10}}{\sum_{i=1}^{m} x_i} \geq 80$$

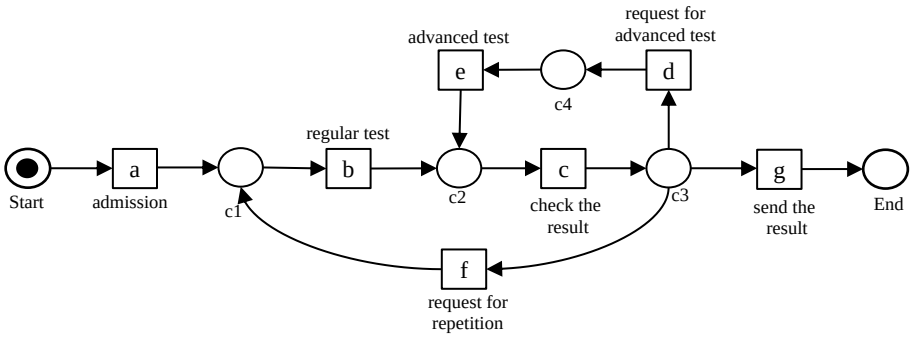
$$\frac{88x_1+88x_2+88x_3+75x_4+63x_5+75x_6+63x_7+50x_8+40x_9+30x_{10}}{\sum_{i=1}^{m} x_i} \geq 78$$

$x_i = 0\ ,\ 1$

The answer of the above problem is unique: $x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = 1$ and $x_7 = x_8 = x_9 = x_{10} = 0$. Therefore, *SelectedCases* = {Patient#1, Patient#2, Patient#3, Patient#4, Patient#5, Patient#6}, leading to the log $\{\langle a, b, c, g\rangle^3, \langle a, b, c, d, e, c, g\rangle^3\}$. For this subset of cases, the aggregation satisfaction level for G1 and G3 will be 81.3 and 79.5, respectively. The α-algorithm produces Model 3 from this log.

6.4.3 Guaranteeing Comprehensive Satisfaction Levels

Here, from a table perspective, the focus is on the *comprehensive* satisfaction level, which we assume to be the *overall* satisfaction level of the *aggregated* levels of all goals. Therefore, the goal model should be used to evaluate s_{Comp} based on the satisfaction levels of all sub goals. To this end, the equation derived from the goal model in Section 6.2 is used to calculate the satisfaction level of the main goal (G_6) as the *comprehensive* satisfaction level.



Model 4. To satisfy goal criteria of Q_{Comp}

- *Organization perspective*: Generate a model where the comprehensive satisfaction level is no less than 75.

The above criterion leads to Q_{Comp}=75. Recall that $s_{Agg.j}$ is the aggregated satisfaction of goal *j*, in our case the *average* of column of $Goal_j$ in Table 15. According to the function derived from the goal model of Figure 24, Algorithm 3 generates the optimization problem as follows:

$$Max\ z = \sum_{i=1}^{10} x_i$$

s.t.

$$x_1 = x_2 = x_3, \qquad x_4 = x_5 = x_6, \qquad x_8 = x_9 \quad \leftarrow (\textit{all-or-none}\ \text{rule})$$

$$Minimum\left(\frac{\sum_{i=1}^{m} x_i.s_{i,4}}{\sum_{i=1}^{m} x_i}\ ,\ 0.4 \times \frac{\sum_{i=1}^{m} x_i.s_{i,1}}{\sum_{i=1}^{m} x_i} + 0.35 \times \frac{\sum_{i=1}^{m} x_i.s_{i,2}}{\sum_{i=1}^{m} x_i} + 0.25 \times \frac{\sum_{i=1}^{m} x_i.s_{i,3}}{\sum_{i=1}^{m} x_i}\right) \geq 75$$

$$x_i = 0, 1$$

($s_{i,j}$ refers to the cells of Table 15, e.g., $s_{2,1}$= 94)

The answer of the above problem is, also, unique: $x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = x_7 = 1$ and $x_8 = x_9 = x_{10} = 0$. Therefore, *SelectedCases* = {Patient#1, Patient#2, Patient#3, Patient#4, Patient#5, Patient#6, Patient#7}, resulting in the log $\{\langle a, b, c, g\rangle^3, \langle a, b, c, d, e, c, g\rangle^3, \langle a, b, c, f, b, c, g\rangle\}$. For this subset, the *comprehensive satisfaction level* is 79.5. The α-algorithm produces Model 4 from this log.

6.5. Discussion

We generated three models using three types of goal-related criteria. Comparing the models, we find that the main difference between Model 2 and the other models relates to the loops. Model 2 does not allow repeating the regular test or to do an advanced test. Here,

the decision point "check the result" is spurious as it will not actually make any decision. According to the goals considered in the generation of such a model, i.e., G1 and G3, one can hypothesize that doing advanced blood test or repeating the regular test are not aligned with the goals of having short process time and of providing all patients with a smooth process. Considering the trace $\langle a, b, c, g \rangle$ that Model 2 can generate, we find that there are three patients who experienced this trace. One of these patients has been diagnosed wrongly ($s_{3,4} = 0$). The goal model shown in Figure 24 implies that the goal "To screen accurately" participates to an AND refinement; therefore, when this goal is denied, regardless of the other goals in the refinement, the main goal of the process gets denied. Using the goal model, the overall satisfaction levels of those three patients are 97, 95, and 0, respectively. This suggests that although Model 2 highly satisfies all cases in terms of process time and smoothness of the process, it may end up with a third of the patients who will be dissatisfied; of course, the sample is very small here, but similar results on a much larger log would improve the confidence in this conclusion.

The above analyses are simple examples of knowledge that GoPED can provide. Such knowledge, together with the discovered models, can help domain experts (re)design goal-aligned process models and encourage good behaviours.

It is also possible to discourage bad behaviours by discovering processes that are worse than unacceptable levels of goal satisfaction. Such processes can then be distributed in organizations as examples of what not to do (i.e., counter-examples, or anti-patterns).

Chapter 7. Experiment

This chapter aims to evaluate the three GoPED algorithms and to analyze their performance. In order to do so, the algorithms were implemented in Python in the *GoPED tool*. As Algorithms 2 and 3 require forming an optimization problem, we also integrated a reputed solver of optimization problems, namely the IBM CPLEX package [181]. In Algorithm 2 and 3, we first adopted the CPLEX syntax to define the optimization problem (using Subject, Constraints, and Variables). Then, our Python programs invoke the CPLEX solver to solve the problem and find the optimal answer. This optimal answer is the subset of cases that should be used as input of process discovery (Figure 22). In this chapter, the preliminaries of experiments are discussed first, then the steps of the experiment and the results are reported.

7.1. Experiment Preliminaries

7.1.1 Main KPI of the Algorithms

In the experiment phase, we use *run time* as a key performance indicator to evaluate the algorithms and analyze the algorithms' sensitivity to different aspects of their input, including the enhanced event log and the goal criteria. Times are measured on an Intel Core i7-2760QM CPU at 2.40 GHz, 8.00 GB RAM, and Windows 7 (an old and slow machine by today's standards).

7.1.2 Factors Considered in the Experiments

Six factors will be considered during the experiments. Those factors are related to either *event logs* or *goal criteria.*

Event log factors

1. Number of cases
2. Number of traces

3. Distribution of cases among traces
4. Length of traces

Goal criteria factors

1. Number of considered goals (for Algorithms 1 and 2)
2. The criteria's boundaries

7.1.3 Experiment Question

When proposing algorithms, the intention is to produce robust algorithms that are useable and scalable to real-world problems. Accordingly, the main goal of the experiment phase is to answer the question below for all three algorithms:

> *Are the algorithms able to select cases from a large (real-world) scale data log within a reasonable time? If so, how does each factor of the event log and goal criteria influence the execution time of the algorithms?*

In order to answer the above question, we need to apply the algorithms on some real-world scale event logs with different values of the four event-log factors mentioned above. In addition, the two remaining factors should be considered to assess the impact of the number of goals and their criteria's boundaries.

7.1.4 One-at-a-Time Rule

The first part of the experiment question can be answered by applying the algorithms on some real event logs (e.g., Sepsis data log discussed in Chapter 8). However, the size of the event logs in real organizations may vary by an order of magnitude. A sensitivity analysis can show how the performance depends on the number of cases, traces, goals, etc. Such an analysis can provide us with a perception about whether we can generalize the algorithms' performance. On the other hand, the sensitivity analysis determines how the run time (as the dependent value) is affected when other variables (factors) change. In order to find the impact of one particular factor on the run time, we run the algorithms over several identical data logs whose only one factor differs from each other (and the other factors are identical). Allowing multiple factors to change makes a tangle of causalities that make it difficult and inaccurate to trace which effect is related to which factor.

7.1.5 Real Data vs. Synthetic Data

Following the *one-at-a-time rule*, we need several data logs whose factors differ by a predefined amount. For example, when experimenting the impact of the number of cases on the running time, we need several event logs whose numbers of cases are different (in a way that is required) while their other factors are similar. Providing such event logs recorded from real-world behaviours is technically possible. As the changing factor should be changing in a specific way that we need for our experiment, the event logs should be generated with respect to our experiment plan.

7.1.6 Challenges in Generating Data Logs for Experiments

Generating some data logs whose factors remain the same except for one factor can be challenging. This challenge relates to the interdependent nature of some factors where changing one of them impacts another one. For example, changing the number of cases without changing the number of traces is possible, but keeping the distribution of cases among the traces can be a challenge. Suppose $L_1 = \{t_1^5, t_2^3, t_3^3\}$; this log has 11 cases and three traces. Generating L_2 with 15 cases and three traces can end up with $L_2 = \{t_1^8, t_2^2, t_3^5\}$ or $L_2 = \{t_1^7, t_2^4, t_3^4\}$ both of which have 15 cases and three traces, but the *distribution of cases among traces* are different. Accordingly, finding one L_2 that can be considered as an event log whose all factors are the same as L_1 except the number of cases is challenging. To deal with this challenge, it is essential to define or quantify the notion of *distribution of cases over the traces* first, and then define a way to generate different even logs with different numbers of cases and the same *distribution of cases among traces.*

On the other hand, because the nature of *the distribution of cases over the traces* is strongly dependent on the *number of traces*, we cannot keep the former constant while the latter changes.

7.1.7 Defining the Notion of Distribution of Cases Among Traces

When generating an event log to be used in the experiment, after defining the number of cases and traces, the question of "*how the cases distribute within the traces"* should be addressed. For example, event log L includes 12 cases and four traces.

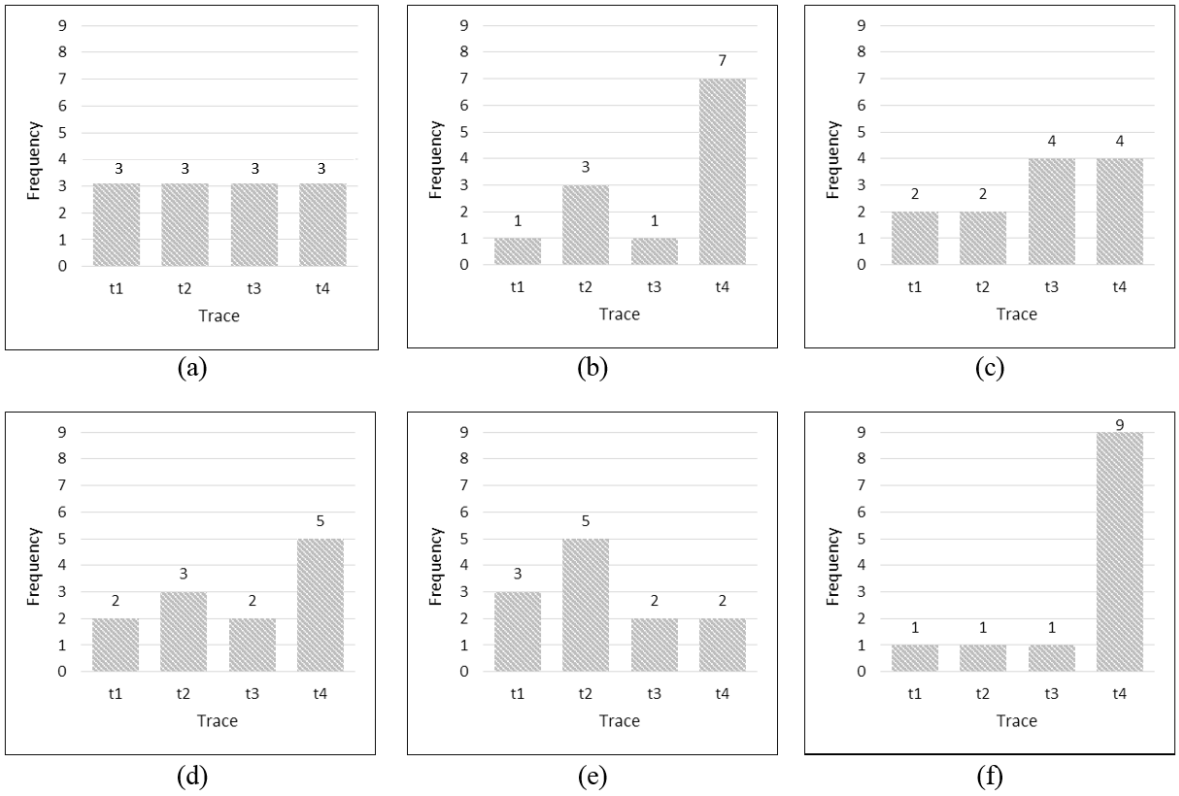


Figure 25. Different distributions of a log with 12 cases and four traces

Figure 25 shows some different ways in which the cases might be distributed among the four traces t_1, t_2, t_3, and t_4. In order to find a factor that reflects the distribution, three assumptions are made:

1. The trace labels (e.g., t_1) do not have any computational value, and we deal with traces as a categorical value whose set of frequencies is considerable. Therefore, in event logs shown in Figure 25, logs (d) and (e) are precisely the same as each other (i.e., both suggest the set of frequency {2, 2, 3, 5}.
2. Regardless of the distribution, the average frequency of traces equals to $\frac{Number\ of\ cases}{Number\ of\ traces}$ (e.g., this average is three for all logs in Figure 25).
3. The frequency of traces can take a value from one to *Number of cases* – *Number of traces* +1 (e.g., for the above example, the frequency of traces can range from one to nine).

Different approaches can be considered to quantify "*distribution*", one of which is the *Variance* of the frequencies of traces. This factor for a pair of given *Number of cases* and *Number of traces* can vary from zero (when all cases are evenly distributed in traces, like Figure 25-a) to $\frac{(Number\ of\ cases - Number\ of\ traces)^2}{Number\ of\ traces}$ (when all traces happened one time except one trace that happened *Number of cases – Number of traces* +1 times. For the above example, the maximum variance possible is 16, which occurs in Figure 25-f). We call such a distribution a *Maximum Variance* (MV) distribution. Because of two drawbacks of *Variance*, that factor was not selected as a function reflecting distribution in our experiment:

a. Variance is not a one-to-one function. There might be many different sets of frequencies with the same mean and variance but quite different distributions.
b. For the experiment, we need to generate some frequencies for traces with intentionally predefined variances (a number between zero and MV, explained above). For example, when there are 5000 cases and 1000 traces, the variance can be considered between zero and 16000. Therefore, to assess the correlation of run time and the variance, some sets of frequencies with the same size of 1000 elements and the variance of 0, 1000, 2000, …, 15000, 16000 are needed. Generating such sets is computationally too complex.

The main goal here is to generate several sets of f_i with different distributions where:

- f_i is the frequency of trace *i*
- $\sum f_i = Number\ of\ cases$
- 1<= f_i <= (*Number of cases – Number of traces +1*)

A useful, heuristic way to generate the different distributions of cases over the traces could benefit from the capabilities of the Beta distribution [182] from probability theory. In probability theory, the Beta distribution is defined over the range between zero and one. There are two parameters, $\alpha, \beta > 0$, which control the shape of the distribution and determine if the distribution has one mode and whether it is symmetrical.

One characteristic of the Beta distribution is that it is supported on a bounded interval. Therefore, it can be used to generate a random variable within a limited range. The Beta distribution can also take a wide range of shapes when its parameters change. It can

approach different distributions when the parameters α and β approach some ranges of value. For example, when α and β are large (e.g., both are greater than three), the density function of Beta can be approximated by the Normal distribution.

Additionally, the Beta distribution with parameters $\alpha = \beta = 1$ is identical to a Uniform distribution. When $(\alpha, \beta) = (2, 1)$ or $(1, 2)$, the distribution makes a triangle. Figure 26 shows the Beta distribution's probability density function (PDF) with different pairs of parameters. As Figure 26 shows, the Beta distribution is highly flexible, and its PDF can be bell-shaped, U-shaped with asymptotic ends, severely skewed, or even straight lines.

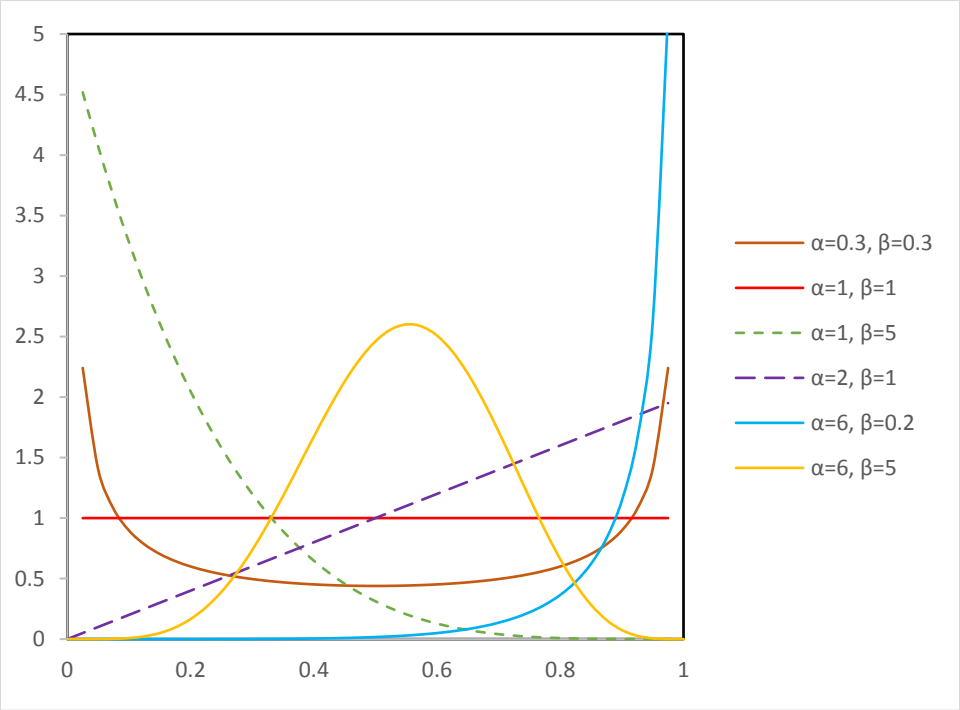


Figure 26. Beta distributions with different parameters

According to the characteristics of the Beta distribution, it can be used as a basis to generate random variables to assign a case to a trace. In this way, we divide the interval [0, 1] into *Number of traces* bins and call these bins $t_1, t_2, \ldots t_{Number\ of\ traces}$. First, a random variable (v) of Beta (α, β) is generated for each case. Then the case is assigned to the trace of the bin that contains *v*. This task will be repeated until all the cases are assigned. In order to generate different logs with the same number of cases and the same number of traces but different distributions of cases among traces, we change the parameters (α, β) and generate a

new log. In this way, the pairs of (α, β) will be defined intentionally to cover different shape families.

Through the way mentioned above, the probability of assigning a given $case_k$ to the trace i (that is represented by bin i) can be calculated as follows:

$$\Pr(Case_k \in Bin_i) = Pr\left((i-1)\cdot\frac{1}{\#\ of\ traces} \le Beta(\alpha,\beta) < i\cdot\frac{1}{\#\ of\ traces}\right)$$
$$= F_X\left(i\cdot\frac{1}{\#\ of\ traces}\right) - F_X\left((i-1)\cdot\frac{1}{\#\ of\ traces}\right)$$

Where:

$1 \le k \le \#of\ cases$

$1 \le i \le \#of\ traces$

$F_x(x)$ is the cumulative distribution function (CDF) of the Beta distribution.

If we call the probability of assigning a given case to trace i as P_i, we are doing as many a number of cases independent experiment, each with its own Boolean-valued outcome: assigning the cases to trace i or not. In this sense, the total number of cases that are assigned to trace i (so called f_i as the frequency of trace i) will be a random variable with binomial distribution with parameters n = number of cases and $p = P_i$. Therefore, the expected value for frequency of trace i equals to *Number of cases* × P_i.

Example. Let say we have 1000 cases and 50 traces. We want to make a log using the Beta distribution with (α =1, β =5) as a distribution of cases among traces. As Figure 27 shows, P_5 is the area under the PDF of the Beta distribution. Therefore, we have:

$$P_5 = Pr\left(4\cdot\frac{1}{50} \le Beta(1,5) < 5\cdot\frac{1}{50}\right) = F_x(\frac{5}{50}) - F_x\left(\frac{4}{50}\right) = 0.40951 - 0.34092 = 0.06859$$

Where: $F_x(x)$ is the CDF of the Beta distribution with (α =1, β =5).

This means that the probability that a random variable with Beta(1, 5) is between 0.08 and 0.1 (as the interval of Bin 5) is 0.06859. As we have 1000 cases, the expected value of the number of cases assigned to $trace_5$ will be 1000×0.06859 = 68.59 cases. This number for $trace_1$ is 96.1 cases and for $trace_{30}$ is 2.83 cases.

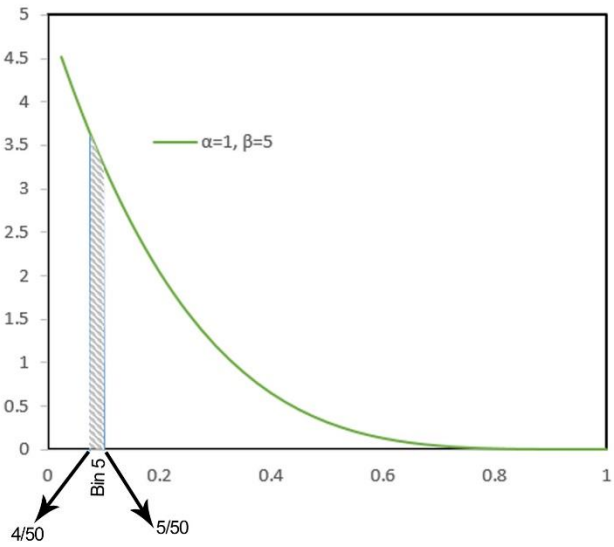


Figure 27. PDF of the Beta distribution with (α =1, β =5).

After generating different distributions, the algorithms will be applied to find the running time.

7.2. Experiments Steps and Results

Following the *one-at-a-time* rule, we consider each of the four event log factors one by one, in the following order:

1. Distribution of cases among traces
2. Number of cases
3. Number of traces
4. Length of traces

7.2.1 Distribution of Cases Among Traces

In this step, the main goal is to test the association between the run time and the distribution of cases among traces.

Method

1. Generating six event logs with the factors below:

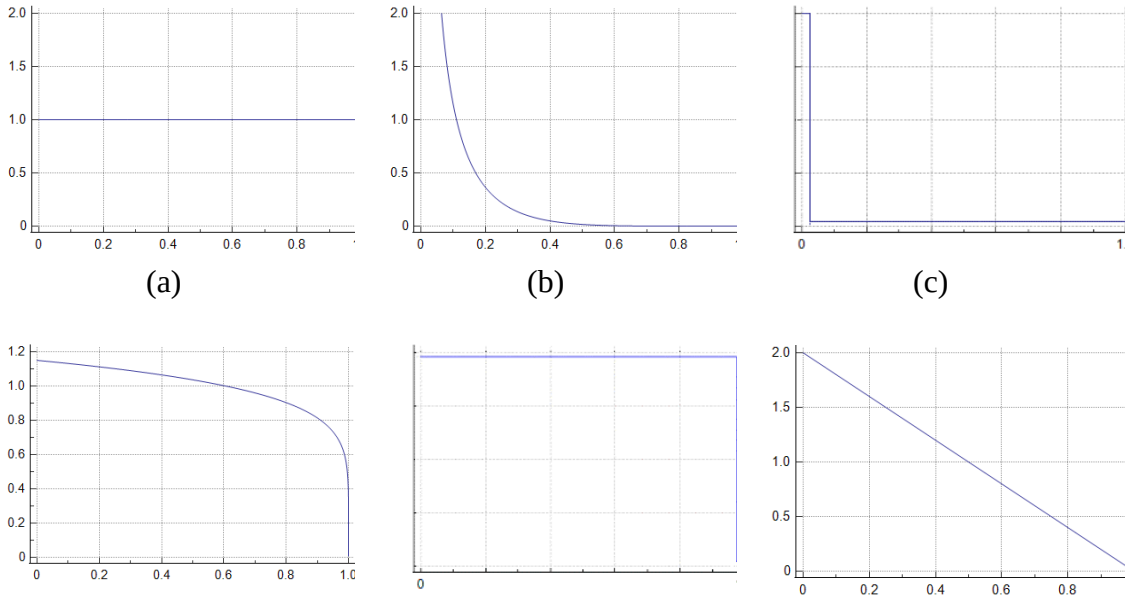


Figure 28. PDFs used for assessing the influence of distribution of cases over traces

- Distribution of cases among traces (the factor that will change):

  Six different typical distributions are used as follows:

  - ***Dist#1*:** Beta(1, 1), which is the same as Uniform distribution. Using this distribution, we aim to assign the same number of cases to each trace (Figure 28-a).

  - ***Dist#2***: Beta(0.2, 6), which is highly skewed, and the distribution makes a concave-up curve. By this distribution, a large number of cases will be assigned to a small number of traces, and a small portion of cases will be assigned to a large portion of traces (Figure 28-b).

  - ***Dist#3***: MV distribution explained in Section 7.1.7. Dis#3 is technically an extreme version of Dist#2 where all traces except one have only a single case, and the remaining trace has all the remaining cases (Figure 28-c).

  - ***Dist#4***: Beta(1, 1.15) where the distribution makes a concave-down curve (Figure 28-d).

- ***Dist#5***: An extreme version of Dist#4 (called Ex4) where one trace has only one case, and all reaming traces are equally assigned to the remaining cases. This distribution represents an extreme version of Dist#4 (Figure 28-e).
- ***Dist#6***: Beta(1, 2), which makes a triangle-shaped distribution. This distribution represents a middling between Dist #2 and Dist#4 (Figure 28-f).

- Number of cases: 50,000
- Number of traces: 1000
- Length of traces: 6 (e.g., t00001)
- Number of considered goals: 2 goals (G1 and G2)
- The criteria's boundaries:

  Algorithm 1: G1≥80% and G2≥75% with 70% confidence level

  Algorithm 2: G1≥80% and G2≥75%

  Algorithm 3: Comp≥70%

2. Applying the algorithms on six generated event logs with the defined goal-related criteria, 20 times.
3. Comparing six groups of running times.

Results

As the charts in Figure 29 show, there is no meaningful trend for the run time when the distribution of cases among traces changes for all three algorithms. It is noteworthy that in these experiments, the statistical correlation is not tested. Instead, the practical correlation that shows a meaningful difference in terms of the feasibility of the algorithm is considerable. For example, based on a one-way ANOVA test with a significance level of 0.05, the null hypothesis of "the means of all six groups are equal" will be rejected for Algorithm 1. That means the six groups of running times do not have the same mean. Practically, the average time of 20 runs for Dist#5 is 0.276 s (as the maximum), while the average time of 20 runs for Dist#3 is 0.248 s (as the minimum). The difference between those two running

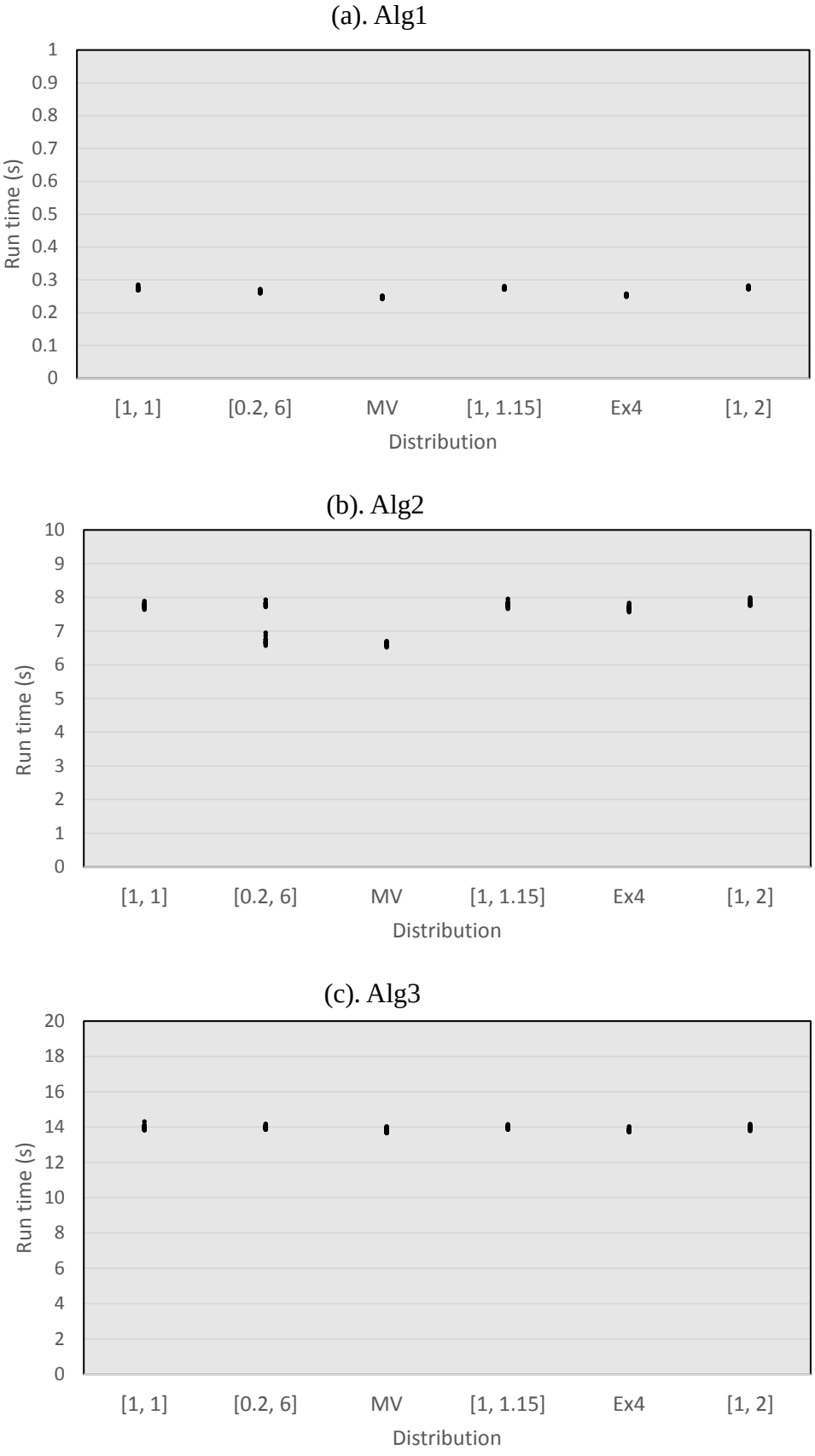


Figure 29. Running times for different distributions of cases among traces.

times is only 11% (i.e., 0.028 s). 11 percent as the maximum range of running time does not suggest a significant correlation between the *running time* and the *distribution of cases among traces*. The same condition holds for the two remaining algorithms as well.

7.2.2 Number of Cases

Here, the main goal is to consider the impact of the number of cases on the run time of the algorithms. As the number of cases shows the size of the data log that the algorithm should process, we anticipate that this factor positively correlates with the run time.

Method

1. Generating ten event logs with the factors below:
    - Number of cases: 50×10^{x} x: 1, 4/3, 5/3, …, i/3, …,4

      Number of cases: 500, 1077, 2321, 5000, 10772, 23208, 50000, 107722, 232079, 500000 (i.e., a logarithmic progression)
    - Number of traces: 500
    - Distribution of cases among traces: Beta(0.2, 6)
    - Length of traces: 6
    - Number of considered goals: 2 goals
    - The criteria's boundaries:

      Algorithm 1: G1≥80% and G2≥75% with 70% confidence level

      Algorithm 2: G1≥80% and G2≥75%

      Algorithm 3: Comp≥70%
2. Applying the algorithms on the generated event logs with the defined goal-related criteria, ten times.
3. Drawing a scatter plot to provide a visual representation.

Results

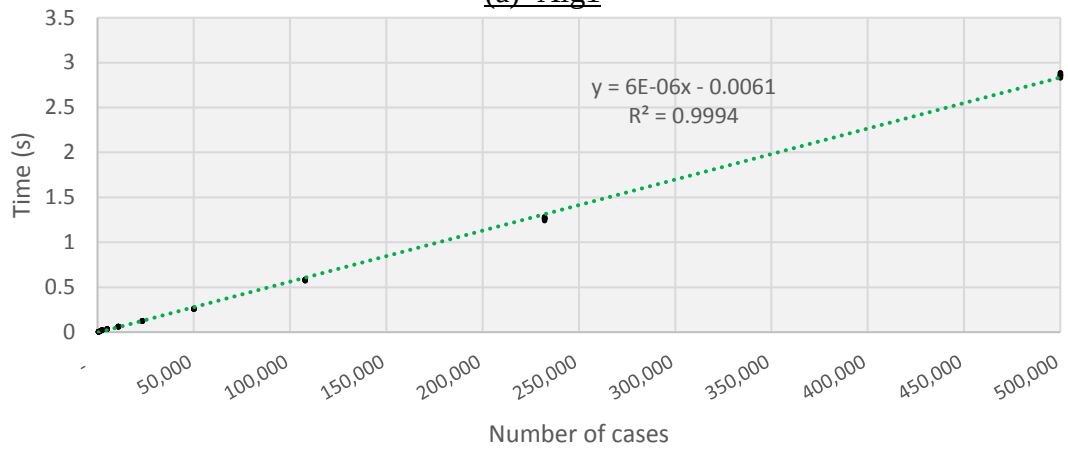
As Figure 30 shows, the running time of all three algorithms increases when the number of cases increases. It is not surprising as the number of cases defines the size of the event log. The first algorithm works based on searching within cases, and the other two algo-

rithms work based on an optimization problem whose number of variables equals the number of cases in the event log. Figure 30.a shows a linear correlation between the number of cases and running time for Algorithm 1. Both variables increase concurrently and at a constant rate. Based on a statistical test, P-value is almost zero, which accepts the regression line's goodness of fit. The coefficient of determination (R-squared) is 0.999, meaning 99.9% of the variance in the running time can be explained by the number of cases.

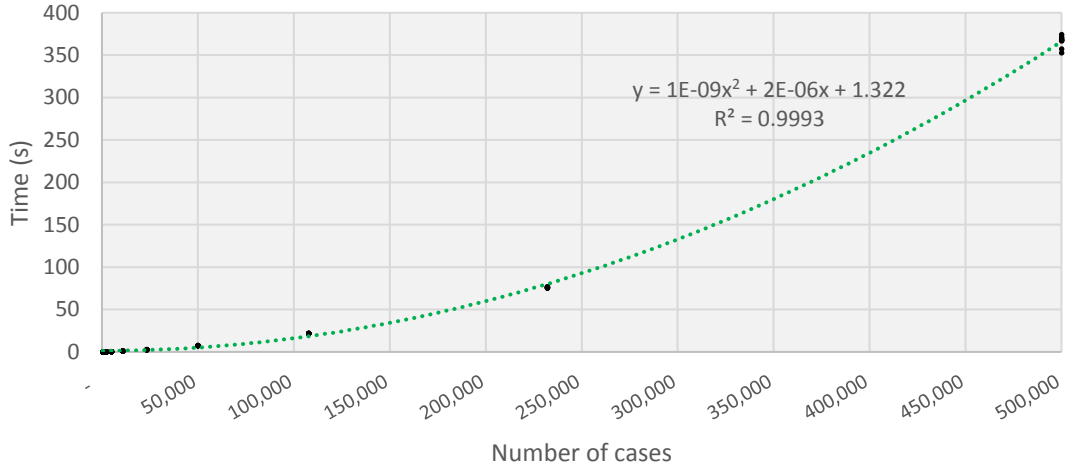
Figure 30.b, shows that a quadratic polynomial trendline is a good fit to the trend of running time when the number of cases increases. The R-squared of 0.999 means that for Algorithm 2, 99.9% of the variance in the running time can be explained by the number of cases. Figure 30.c shows that for Algorithm 3 a similar positive correlation applies.

Comparing the maximum running time of each algorithm that happens when the number of cases is 500,000 reveals that with the same event log, the running time of Algorithm 3 is about two times longer than Algorithm 2 and about 300 times longer than Algorithm 1. Algorithm 1 is a straightforward algorithm that uses some loops, but the other two algorithms employ CPLEX to solve an optimization problem. Accordingly, it is not surprising that Algorithm 1 works way faster than two other algorithms. However, for Algorithms 2 and 3, the running time is defined by the optimization problem's features. The number of variables and the number of constraints are two factors that should be considered here. In fact, when the number of cases is 500,000, and the number of traces is 500, CPLEX generates a problem with 500,004 variables and 499,515 constraints for Algorithm 3 and 500,000 variables and 499,502 constraints for Algorithm 2. Algorithm 3 should deal with the Max and Min functions, coming from the goal model, in the constraints. Therefore, some dummy variables will be added to the problem to convert the Max and Min functions to linear constraints. That is why Algorithm 3 made 13 variables and four constraints more than Algorithm 2. However, four variables and 13 constraints might not be the source of that longer running time due to the total number of these factors (i.e., 500,000). As this topic is related to the performance analysis of optimization problem solvers and is out of the scope of this thesis, we avoid considering it any longer.

(a)- Alg1



(b)- Alg2



(c)- Alg3

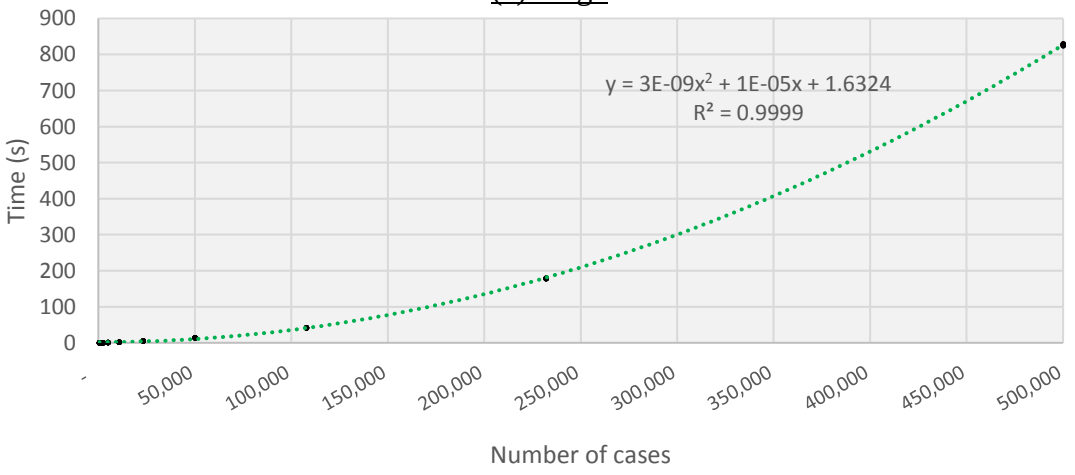


Figure 30. Running times for different numbers of cases

7.2.3 Number of Traces

The number of traces might statistically have a significant impact on the running time of the algorithms. In this step, the main goal is to find how the *running time* changes when the number of traces changes.

Method:

1. Generating 11 event logs with the factors below:
   - Number of cases: 50,000
   - Number of traces: $1 + i \times \frac{50000-1}{9}$ i=0, 1, …, 9

     Number of traces: 1, 5556, 11112, 16667, 22223, 27778, 33334, 38889, 44445, 50000 (a linear scale)
   - Distribution of cases among traces: Beta(0.2, 6)
   - Length of traces: 6
   - Number of considered goals: 2 goals
   - The criteria's boundaries:

     Algorithm 1: G1≥80% and G2≥75% with 70% confidence level

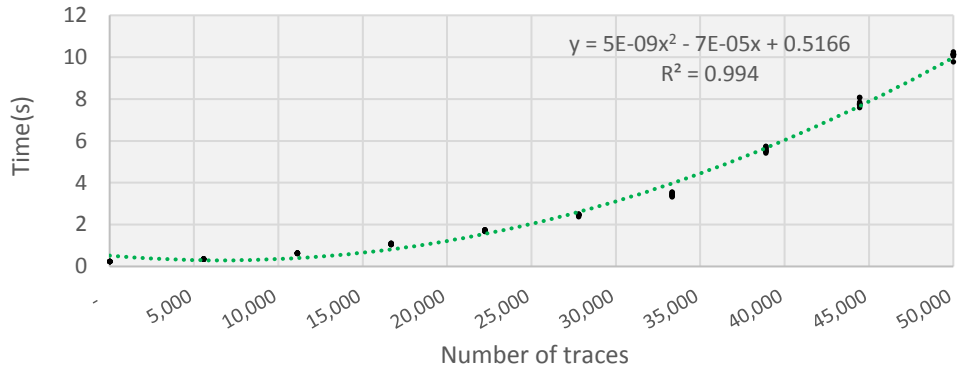
     Algorithm 2: G1≥80% and G2≥75%

     Algorithm 3: Comp≥70%
2. Applying the algorithms on the generated event logs with the defined goal-related criteria, ten times.
3. Drawing a scatter plot to provide a visual representation.

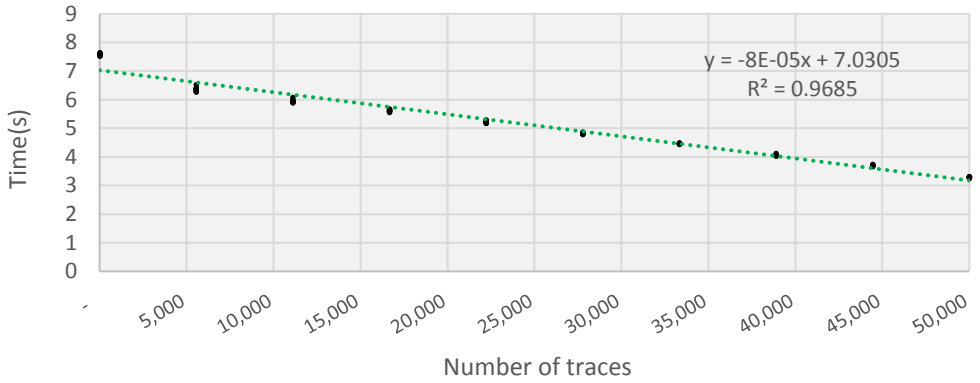
Results

As Figure 31.a shows, the running time of Algorithm 1 has a strong quadratic correlation with the number of traces with a very strong R-square=0.99. While the correlation between the two variables is positive for Algorithm 1, for the second algorithm, the correlation is negative. As Figure 31.b shows, a linear trendline with a negative slope with a high R-square suggests that the running time *decreases* when the number of traces increases. For Algorithm 3, using all running times with an R-square of 0.38 does not suggest a good fit trendline that shows a strong correlation (Figure 31.b). However, if the average of 10 running times was used as a running time for each number of traces, a linear correlation

(a). Alg1



(b). Alg2



(c). Alg3

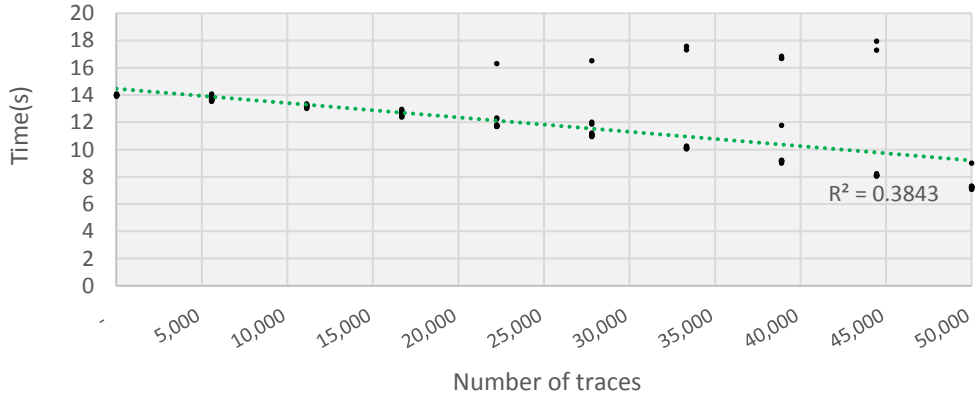


Figure 31. Running times for different numbers of traces

with a negative slope and R-square of 0.83 will be resulting. Therefore, we can conclude that the running time of Algorithm 3 decreases when the number of traces increases. Nevertheless, the number of traces explains 80% of the variance of the running time.

The negative trend slopes for Algorithms 2 and 3 are not surprising because when the number of traces increases, the number of cases with the same trace decreases. When the number of cases that share the same trace decreases, the number of constraints made to handle the *all-or-none* rule decreases. It is noteworthy that for every *n* cases with the same trace, *n-1* constraints will be generated.

7.2.4 Length of Traces

Traces are the sequences of activities recorded in databases. GoPED algorithms deal with the traces as string variables. In all three algorithms, the log should be sorted based on traces and there are some lines where the traces are compared (e.g., Algorithm 1: lines 1 and 15, Algorithm 2: line 1, and Algorithm 3: line 5). Typically, the time needed for comparing and sorting strings depends on the number of characters that should be scanned. On the other hand, in real-world data logs, the traces are usually long. In this step, the main goal is to find how the *run time* changes when the length of traces changes.

Method

1. Generating ten event logs with the factors below:
   - Number of cases: 50,000
   - Number of traces: 5,000
   - Distribution of cases among traces: Beta(0.2, 6)
   - Length of traces: $6 + i \times \frac{50000-6}{9}$ i=0, 1, …, 9

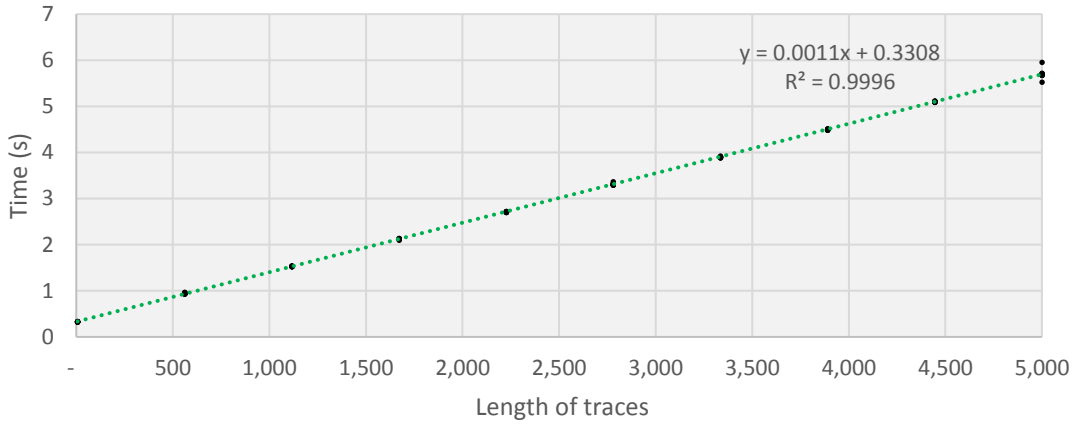
     Length of traces: 6, 561, 1116, 1671, 2226, 2780, 3335, 3890, 4445, 5000
   - Number of considered goals: 2
   - The criteria's boundaries:

     Algorithm 1: G1≥80% and G2≥75% with 70% confidence level

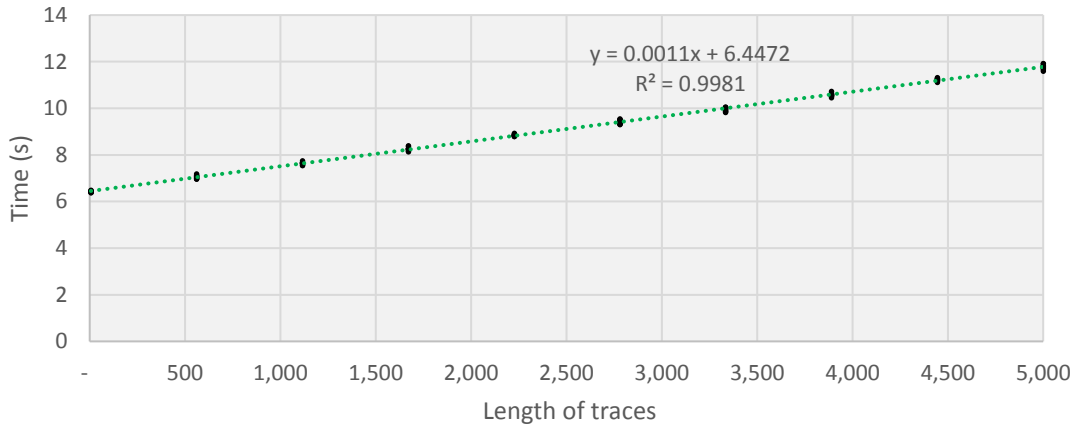
     Algorithm 2: G1≥80% and G2≥75%

     Algorithm 3: Comp≥70%

(a). Alg1



(b). Alg2



(c). Alg3

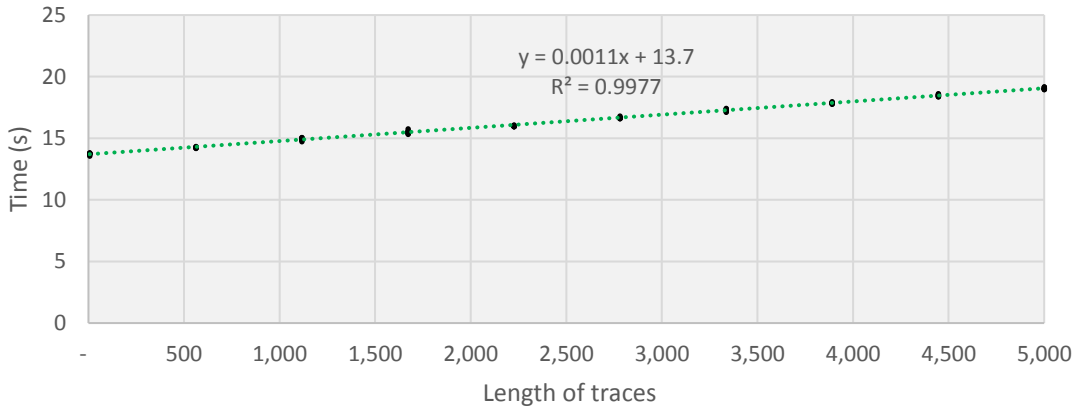


Figure 32. Running times for different lengths of traces

2. Applying the algorithms on the generated event logs with the defined goal-related criteria ten times.
3. Drawing a scatter plot to provide a visual representation.

Results

As Figure 32 shows, there is a strong linear correlation between the running time and the length of traces for all three algorithms. The R-square is close to one for all algorithms and the correlation is positive. Therefore, the running time increases when the length of traces increases. It is noteworthy that the length of traces does not impact the dimensions of the optimization problem generated for Algorithms 2 and 3. The only lines of the three algorithms that are impacted by the length of traces are where the machine wants to perform the method sort() and perform a *comparison* between two traces. List sort() has a runtime complexity of O(n log(n)), and string comparisons do a linear scan of the characters with a time complexity of O(n).

7.2.5 Number of Considered Goals

In Algorithms 1 and 2, a set of goals with corresponding criteria are input. In Algorithm 1, the satisfaction level of each case will be checked against the criteria. Therefore, for each case, the number of comparisons equals the number of considered goals. In Algorithm 2, each considered goal makes a new constraint for the optimization problem. Therefore, it makes sense to consider the impact of the number of considered goals on the run time.

In Algorithms 3, the goal model is used to calculate the comprehensive satisfaction level from aggregated satisfaction level of all considered goals. Therefore, the number of considered goals for this algorithm represents the size of the goal model.

Method

1. Generating 10 event logs with the factors below:
    - Number of cases: 50,000
    - Number of traces: 25,000
    - Distribution of cases among traces: Beta(1, 5)
    - Length of traces: 6

- Number of considered goals: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
- The criteria's boundaries:

  Algorithm 1: 80% for all considered goals, 70% confidence level

  Algorithm 2: 80% for all considered goals

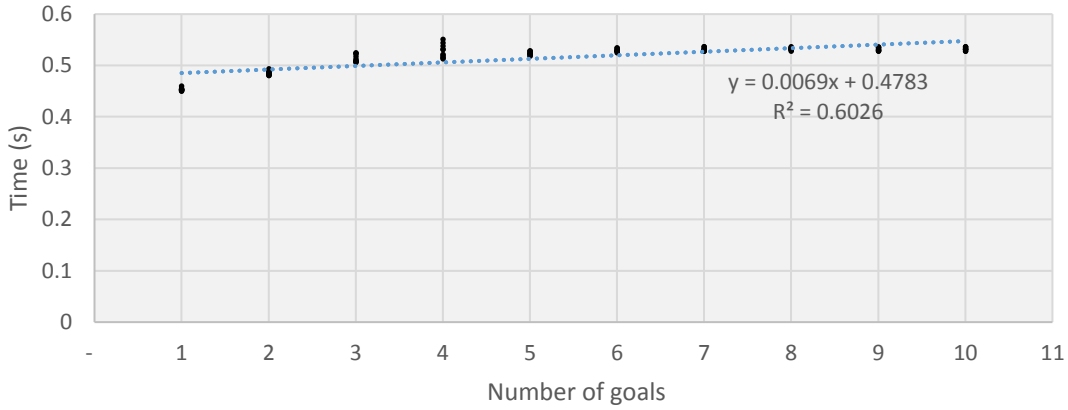
  Algorithm 3: 80% for the compressive satisfaction level

2. Applying the algorithms on the generated event logs with the defined goal-related criteria ten times.
3. Drawing a scatter plot to provide a visual representation.

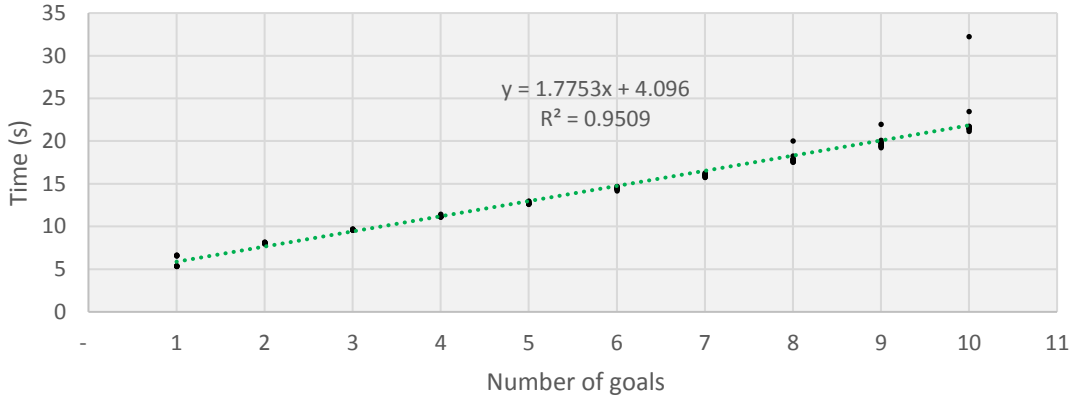
Results

Figure 33.a suggests a very low coefficient (0.0006) for the trendline that practically, with the usual number of goals, will not significantly increase the running time. However, when the number of considered goals increases, Algorithm 1 should make more comparisons for each case. Also, checking the criteria for each case will return false at the first considered goal whose satisfaction level does meet the criterion. Meanwhile, Figure 33.b,c suggest a positive strong linear correlation between the two variables. Therefore, increasing the number of considered goals does not significantly impact the running time of Algorithm 1, but it increases the running times of the two other Algorithms.

(a). Alg1



(b). Alg2



(c). Alg3

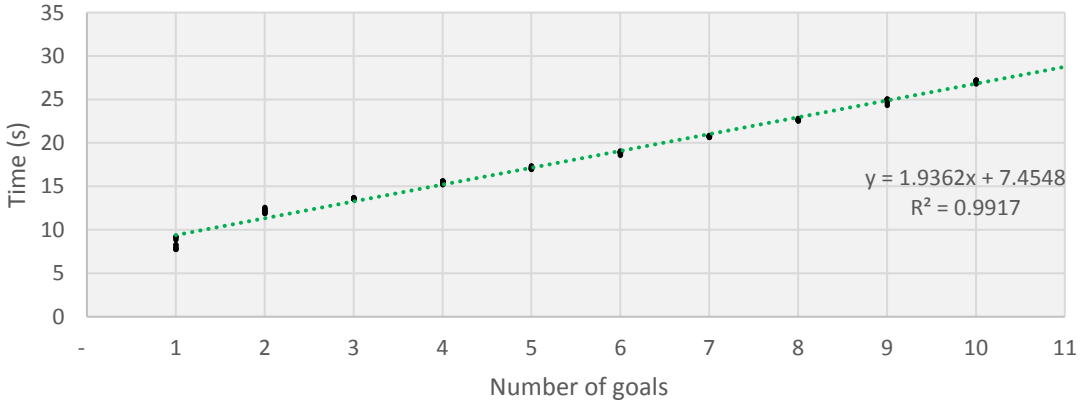


Figure 33. Running times for different numbers of considered goals

7.2.6 Criteria's Boundaries

In this step, the criteria's boundaries are changed to assess whether the boundaries that mathematically impact the number of selected cases will also impact the run time or not.

Method

1. Generating ten event logs with the factors below:
    - Number of cases: 50,000
    - Number of traces: 1000
    - Distribution of cases among traces: Beta(0.2, 6)
    - Length of traces: 6
    - Number of considered goals: 2 goals
    - The criteria's boundaries:

        (1) Algorithm 1, 2 and 3: $1 + i \times \frac{100-1}{9}$ i=0, 1, …, 9

        1%, 12%, 23%, 34%, 45%, 56%, 67%, 78%, 89%, 100% for (both goals)

        (2) Algorithm 1, Confidence level: $1 + i \times \frac{100-1}{9}$ i=0, 1, …, 9

        1%, 12%, 23%, 34%, 45%, 56%, 67%, 78%, 89%, 100%

2. Applying the algorithms on the generated event logs with the defined goal-related criteria ten times.
3. Drawing a scatter plot to provide a visual representation.

Results

As Figure 34 shows, none of the charts suggest a meaningful trend for running times when the threshold for the boundaries increases. There are two points in Figure 34.a and Figure 34.b where the trend of running time changed. Based on an analysis on the data logs, these phenomena happen in the neighborhood of the average of the random distribution function that generated the synthetic satisfaction levels. Therefore, these phenomena are caused by a data bias.

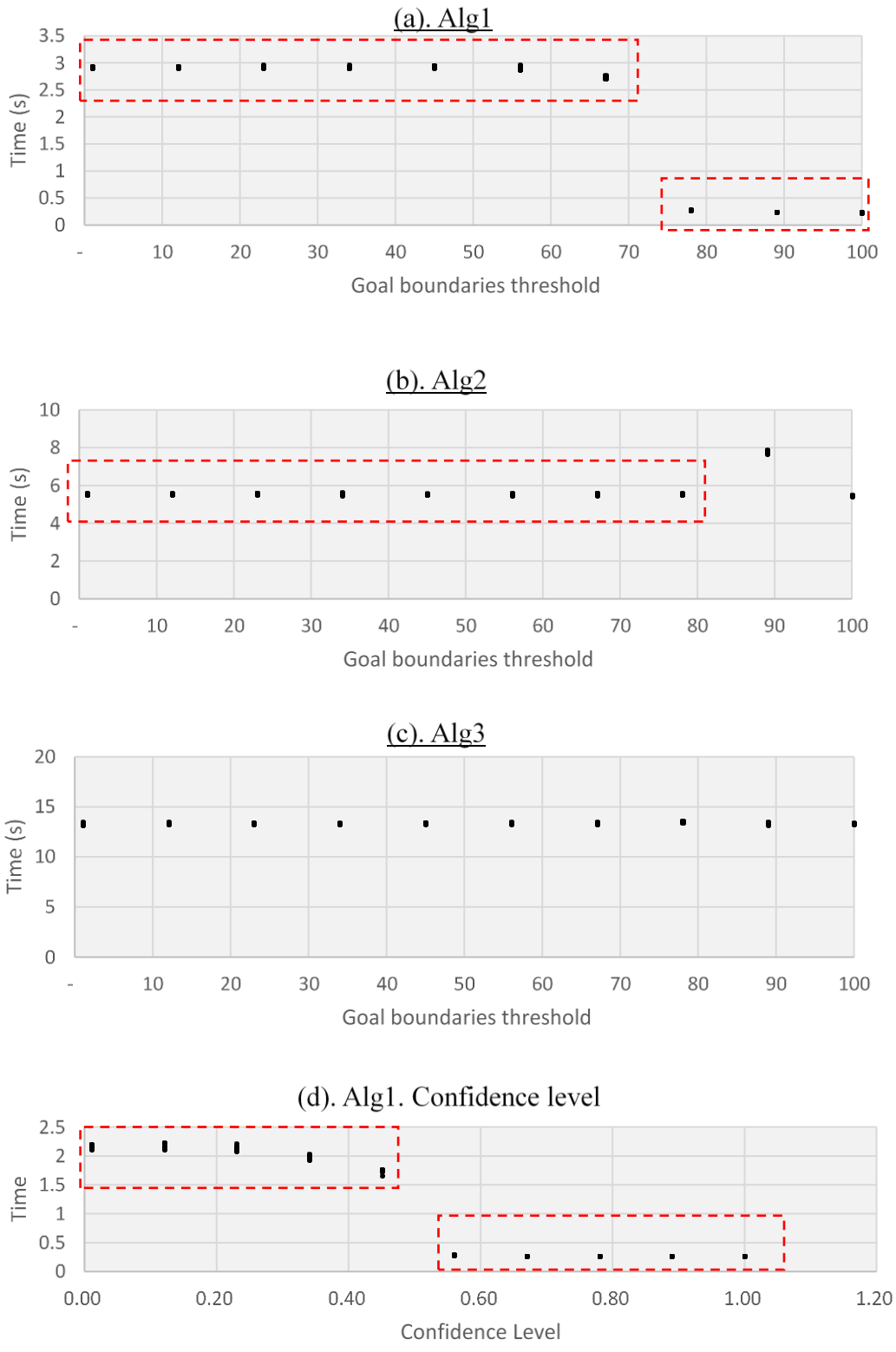


Figure 34. Running times for different criteria's boundaries

The same situation and the same bias are shown in Figure 34.d for the relation between running time and the confidence level for Algorithm 1. The chart suggests that out of the neighbourhood of the mean of the functions that generated the satisfaction level, there is no meaningful trend for the running time when the confidence level changes.

7.3. Discussion

In this chapter, the performance of the GoPED algorithms was assessed in total on 1960 executions of three algorithms on logs of different controlled characteristics. Six factors were considered to find how the running time change when the factors change. Table 16 summarizes the correlation between the running time of the GoPED tool and the six considered factors.

Table 16. Correlation between running time and different factors of event logs

	Correlation between running time and the considered factor		
Factor	*Algorithm 1*	*Algorithm 2*	*Algorithm 3*
Distribution of cases among traces	No correlation	No correlation	No correlation
Number of cases	Positive. Linear	Positive. Quadratic	Positive. Quadratic
Number of traces	Positive. Quadratic	Negative. Linear	Negative. Linear
Length of traces	Positive. Linear	Positive. Linear	Positive. Linear
Number of considered goals	Positive. Linear	Positive. Linear	Positive. Linear
The criteria's boundaries	No correlation	No correlation	No correlation

According to Table 16, the factor that has the most substantial impact on the running time is *the number of cases*. This factor is the only factor that has a positive quadratic correlation with the running time for two Algorithms (2 and 3). Such algorithms select cases by forming and solving an optimization problem.

The longest running time of all 1960 runs is 830 seconds, for the execution of Algorithm 3 on an event log of 500,000 cases using a machine equipped with an Intel Core i7-2760QM CPU at 2.40 GHz, 8.00 GB RAM, and Windows 7. Such a computer, with a CPU from 2013 and an old operating system, can nowadays be outperformed by a factor of 5 on common laptops. The experiments' running times show that the GoPED algorithms are feasible and scalable enough to apply to large event logs, and hence suggest that the algorithms and their implementation can be used in practice.

The next chapter applies the algorithms and corresponding tools to a healthcare-related case study with a real open-source event log. A real process mining tool (ProM) is also used to produce process models after the application of the algorithms.

Chapter 8. Case Study

The previous chapter used experiments to assess the feasibility and scalability of the GoPED method applied to realistic but synthetic data. This chapter aims to further evaluate the GoPED method against a real data log from the healthcare domain, with further connection to process mining tools.

8.1. Evaluation Method

Design Science Research suggests five types of evaluations of designed artifacts: observational (case study, field study), analytical, experimental (controlled experiment, simulation), testing (functional, structural), and descriptive [4]. In this research, in addition to an ongoing example used throughout the thesis chapters to illustrate the GoPED method, the evaluation plan consists of three steps:

1. Prototype tool (Excel-based and programmed) to select traces in a complex log according to specified goal-oriented quality criteria (reported in Chapter 7).
2. Experiments where the event logs were generated considering six major factors that can impact the performance of the methods (reported in Chapter 7).
3. Case study: A healthcare-oriented case study with actual data and real users (this chapter). To this end, the method will be applied on the event log of trajectories of patients with sepsis in a Dutch hospital in The Netherlands.

The detailed specs of considered event log are described first, and then how the goal-related attributes are derived is explained. Three different sets of goal-related criteria were defined in consultation with a domain expert (a physician) and a published paper whose focus is on the same event log [183].

8.2. Sepsis Event Log

The sepsis event log [184] used for evaluation is the events of the journeys of 1050 patients with sepsis recorded by an information system of a Dutch hospital.

Sepsis is a life-threatening condition typically caused by an infection and the presence of harmful bacteria in the blood or other tissues. The body's response to the presence of that bacteria can potentially lead to the malfunctioning of various organs, shock, and death.

The hospital where the sepsis event log was recorded has around 700 beds and about 50,000 patients visit it each year. The scope of the event log is on patients admitted to the emergency ward. The event log focuses on the group of patients for which sepsis treatment is to be expected. The log includes the events stored related to patients from their registration in the emergency room until their discharge. The event log was enriched with data from laboratory test results and triage checklists. The sepsis raw file is a CSV file consisting of 31 columns and 15,125 rows. The first four columns are Case, Event, Start Time, and Complete Time. The remaining 27 columns contain the information of the checklists and test results. As the data of two columns Start Time, and Complete Time are identical, the column Start Time was removed, and the column Complete Time was renamed to Time. Also, the first column was renamed to Case ID and the second column was renamed from Event to Activity. The Case ID of all patients was a combination of some characters (e.g., A, BA) in the original file. To make them easier to read, they were changed to the format "Case####" from Case0001 to Case1050.

The main characteristics of the sepsis event log are as follows:

- Number of events: 15,124
- Number of activities: 16
- Number of cases: 1,050
- Number of unique traces: 846

Using the TraceMaker tool, explained in Section 4.2.4, the table that contains the trace that each patient (case) has followed in the hospital was generated (file name: SepsisTraces). According to the distribution of cases over the traces discussed in Section 7.2.1, as Figure 35 shows, the sepsis event log is highly skewed to the right. There are many infrequent traces that have happened with very few patients. In this log, there are 784 traces (83% of unique traces) that happened only once and there are only five traces that have been followed by more than 10 patients.

As the Dashboard of ProM shows in Figure 36, there is a total of 16 activities performed in sepsis traces. The length of the traces (i.e., the number of activities performed in a trace) ranges from three to 185 activities. Traces in the sepsis event log have 14 activities on average. Each case has at least three unique activities and at most 12 unique activities. As ProM indicates, there are six activities that show up as the first activity of a trace and 14 activities that end a trace.

Table 17 shows the name of all activities performed in the sepsis process together with their frequency of occurrence. As the table shows, there are 10 activities that are executed for more than 60% of patients, and 4 activities that are rarely performed, i.e., for less than six percent of patients. All 16 activities can be grouped into six categories:

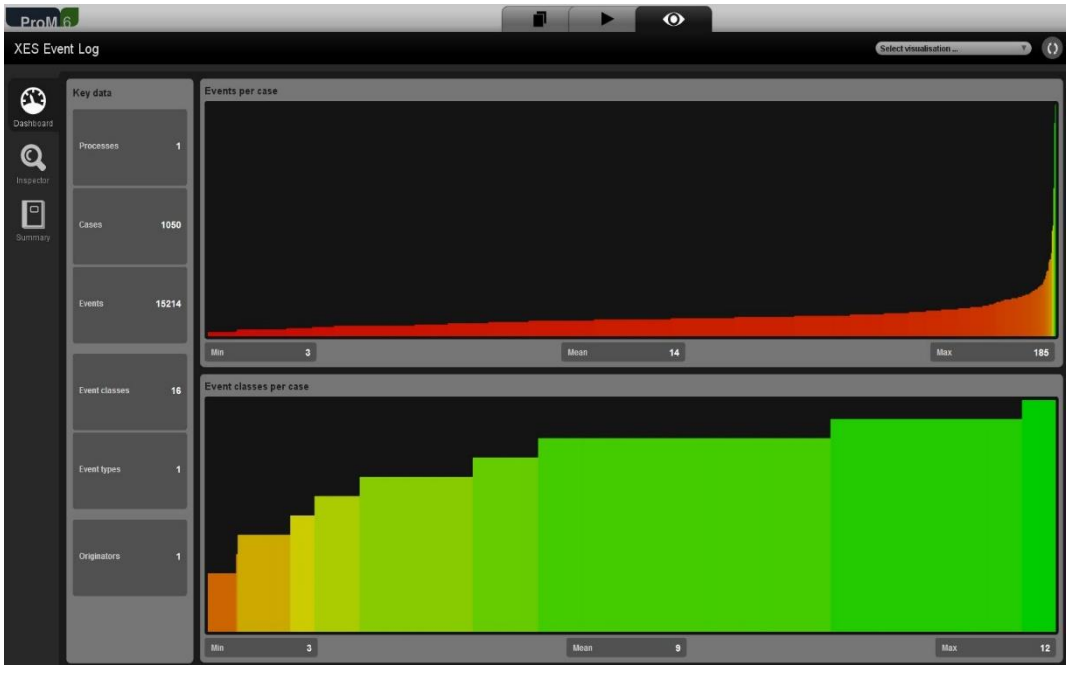


Figure 36. Dashboard of ProM for the sepsis event log

- three activities about registration and triage in the emergency ward;
- three activities regarding the medical test for measuring leukocytes, C-reactive protein (CRP), and lactic acid;
- two activities related to the use of intravenous (IV) antibiotics or other liquids;
- two activities regarding admission to normal care (NC) or intensive care (IC);
- five activities about different variants of discharge (release) from the hospital; and
- one activity about returning patients.

8.3. Process Models Discovered from the Original Log

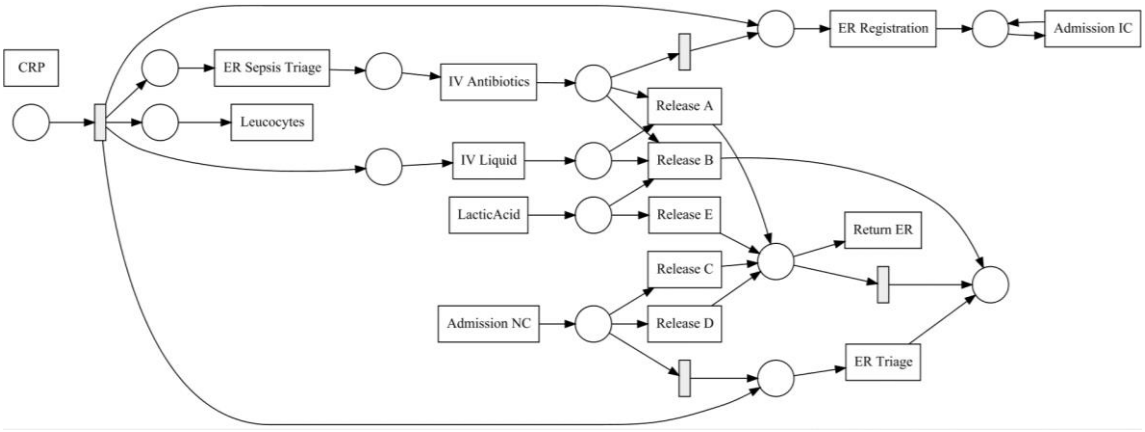
In this section, the sepsis event log is used to discover the underlying process model using different discovery algorithms. Such underlying process models are discovered as a baseline, regardless of any goal-related criteria. Then, the GoPED algorithms will be employed to generate models based on the goal criteria, and the comparison of the models will facilitate a way to evaluate GoPED method.

Table 17. Activities performed in the sepsis event log

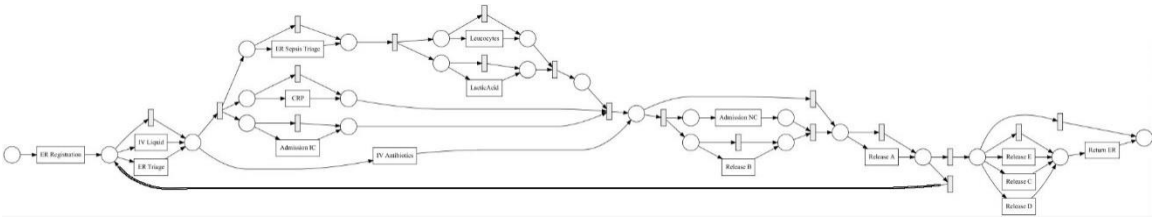
	Activity	*Occurrences (absolute)*	*Cases with this Activity*	
			Count	*%*
1	ER Registration	1050	1050	100.0%
2	ER Triage	1053	1050	100.0%
3	ER Sepsis Triage	1049	1049	99.9%
4	Leucocytes	3383	1012	96.4%
5	CRP	3262	1007	95.9%
6	LacticAcid	1466	860	81.9%
7	IV Antibiotics	823	823	78.4%
8	Admission NC	1182	800	76.2%
9	IV Liquid	753	753	71.7%
10	Release A	671	671	63.9%
11	Return ER	294	294	28.0%
12	Admission IC	117	110	10.5%
13	Release B	56	56	5.3%
14	Release C	25	25	2.4%
15	Release D	24	24	2.3%
16	Release E	6	6	0.6%
	Grand Total	**15214**		

The Alpha algorithm is first used as an algorithm that does not take into consideration the frequencies of activities. Model 3 is the process model resulting from the Alpha miner plugin in ProM. The structure of the discovered model is difficult to read and does not represent a meaningful end-to-end model expected from the sepsis process, such as starting from patient registration (*ER Registration*) and ending with some variants of discharge (*Release A-E*). An isolated activity CRP disconnected from all other activities is also another drawback of that process model.

The second algorithm used to discover the model is Inductive Miner (IM) [185, 186], as an algorithm able to cope with infrequent behaviours and large event logs, while promoting the soundness of the model. The inductive process discovery approach is one of the leading approaches because of its flexibility, formal guarantees, and scalability [1]. The plugin of ProM 6.6 called "*Mine Petri net with Inductive Miner*" was applied on the raw sepsis event log. The plugin was configured to use the *Inductive Miner—infrequent* algorithm (IMF, [186]) with 20% as the *Noise threshold*. The IMF algorithm, as one variation of the basic IM algorithm, uses various types of filtering to show the mainstream behaviour. The IMF algorithm considers the frequency of activities and the frequency of directly-following relations between the activities. IMF is more sophisticated than basic IM and has some similarities with the heuristic miner. Model 4 is the process model returned by IMF. The resulting process model starts with patient registration (*ER Registration*) and ends with different variants of discharge (*Release A-E*). However, most of the activities that occur in between the start and end activities are modelled very imprecisely. Almost for all activities, there is a parallel silent activity that enables skipping the real activity. There is also an arc during the model (highlighted in Model 4) that allow repeating almost all medical activities in the process. The resulting process model returned by IMF is not easily usable by healthcare practitioners such as physicians and nurses. This result is aligned with what Mannhard and Blinde [183] found, when applying IM on the same sepsis event log. Changing the *Noise threshold* for the IMF plugin did not help.



Model 3. Sepsis process model discovered by the Alpha miner plugin of ProM



Model 4. Sepsis process model discovered by the Inductive miner plugin of ProM

8.4. GoPED on the Sepsis Event Log

The models discovered using the Alpha and IMF algorithms resulted from an activity-based approach, where the sequence of activities and/or their frequencies were taken into account. In this section, the value of some KPIs driven from particular medical guidelines [183], after consultation with a physician, are used to calculate their corresponding goals. Then, the sepsis event log will be enhanced by adding the satisfaction levels of the goals. The enhanced event log is the starting point of applying GoPED to the event log.

The GoPED method relies on goals, their relations constructing a goal model, and their satisfaction levels. In addition to the main attributes of the sepsis event logs (Case ID, Activity, and Time), there are 27 columns that show the medical checklists values, medical test results, and the age of the patients. This log, however, does not have any goal-related

attribute explicitly associated to a goal model. Coping with such common challenge, three particular medical guidelines that should be followed within the treatment of sepsis patients [183] are used here as the basis of three KPIs. This approach was discussed with a physician from the University of Ottawa as a domain expert to avoid having attributes (including the medical information stored in 27 additional columns) used as KPIs in an incorrect way.

The medical guidelines for the treatment of sepsis patients require that:

1. patients should be administered antibiotics within one hour,
2. lactic acid measurements should be done within three hours;

According to the aforementioned time rules, two goals and their related KPIs are defined:

- KPI_1 : The time between *ER Sepsis Triage* and *IV Antibiotics*
- KPI_2: The time between *ER Sepsis Triage* and *LacticAcid*

$Goal_1$ is also defined as "To decrease the time between *ER Sepsis Triage* and *IV Antibiotics"*. Likewise, $Goal_2$ is defined as "To decrease the time between *ER Sepsis Triage* and *LacticAcid"*.

The third goal, $Goal_3$, is related the common goal of any healthcare system that is to minimize the number of patients that return to the emergency room of the hospital after they are discharged. There is an activity in the sepsis event log concerned with returning patients later. For 27.8% of the patient, such an activity has been performed, indicating that they have returned to the hospital after 81 days on average. The third KPI, therefore, can be defined as:

- KPI_3: Did the patient return to the ER after discharge? (0 for No and 1 for Yes)

$Goal_3$ is defined as "To avoid returning to the hospital after discharge", with KPI_3 as its sole contributor.

8.5. Enhanced Log of Sepsis Process

After explaining the goal models and the KPIs relate to each goal, the GoPED approach was applied to the original sepsis event log. To this end, four tools (explained in Chapter 4 and Chapter 7) were used on the raw CSV file. First, TraceMaker was used to generate the table that represents the trace that each patient has followed in the hospital. Then, the value

of each KPIs was calculated using the time of the corresponding activities. For KPI_1 and KPI_2, the time of the activities of *ER Sepsis Triage*, *IV Antibiotics*, *and LacticAcid* were used. The *current value* of each KPI was calculated as follows:

KPI_1= time of *IV Antibiotics* - time of *ER Sepsis Triage*

KPI_2= time of *LacticAcid* - time of *ER Sepsis Triage*

KPI_3= 1, if activity of *Return ER* presents in the trace, 0 otherwise

KPI_1 and KPI_2 rely on the timestamp of occurrence of some activities. Therefore, if at least one of the activities is not performed during a trace of a case, the calculation of the corresponding KPI will not be possible. As Table 17 shows, the activity *ER Sepsis Triage* shows up for 99.99 percent of the patients, while the activity *IV Antibiotics* is performed for 78.4% of the patients (i.e., 823 patients). These activities are performed once per patient for the 823 patients. The activity *LacticAcid* is also executed for 81.9% of the patients (i.e., 860 patients). Unlike the two first activities, activity *LacticAcid* occurred more than once in some of the traces. According to the definition of KPI_2 about the time between the occurrence of the activity and *ER Sepsis Triage*, the earliest occurrence of the activity *LacticAcid* was considered.

There are 227 traces where the activity *ER Sepsis Triage* has happened without the activity *IV Antibiotics* eventually happening afterwards. Regarding the notion of the corre-

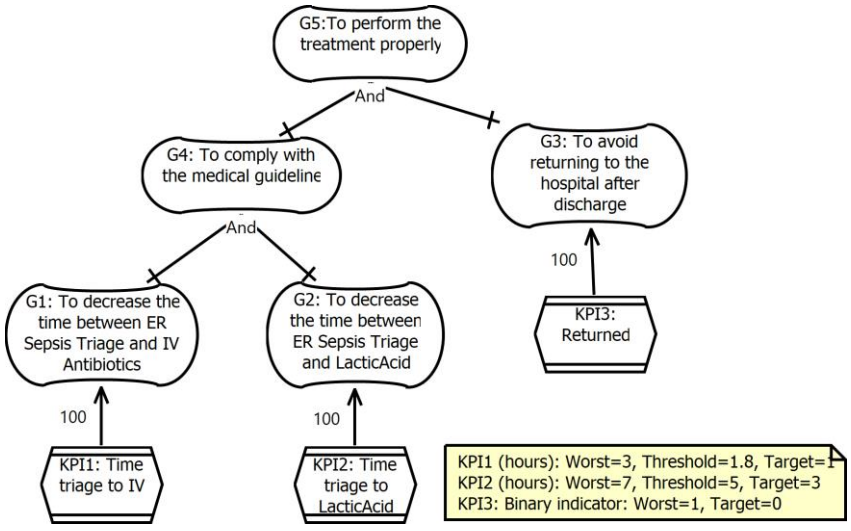


Figure 37. Goal model associated with the sepsis event log.

Table 18. The statistical measurement of the current values of KPI_1 and KPI_2

KPI	*Min*	*Max*	*Mean*	*Median*
KPI_1	0.00 h	26.81 h	1.86 h	1.53 h
KPI_2	0.003 h	398.18 h	1.10 h	0.24 h

sponding rule of the guideline, it was decided that in such cases, the $Goal_1$ would be considered fully dissatisfied (i.e., its level of satisfaction is zero). The same assumption was made for the 190 traces where *ER Sepsis Triage* occurred but *LacticAcid* was absent.

There are 294 patients who have returned to the hospital after discharge and for whom the activity *Return ER* is present in their trace. The KPI_3 used the value 1 for such cases.

Table 18 shows the statistical factors of the current value of the first two KPIs. This table can make an image of the whole event logs in terms of the state of the KPIs in the process. As calculating the satisfaction level of the corresponding goals needs the values of the worst, threshold, and target parameters of these KPIs, these three values were first determined.

For KPI_1, according to the medical guideline, the target value can be assumed to be one hour. That means that any trace where the activity *IV Antibiotics* has performed not later than one hour after the activity *ER Sepsis Triage* has fully satisfied $Goal_1$. As there are no clear threshold and worst values defined by the guideline, it was assumed that the threshold is 1.8 hour (about the average of the current values of the KPI). In addition, the worst value was assumed to be 3 hours.

For KPI_2, based on the medical guideline, the target value was set to three hours. Therefore, if the activity *LacticAcid* was performed no later than three hours after the activity *ER Sepsis Triage*, then $Goal_2$ is fully satisfied. It was assumed that the threshold of KPI_2 is 5 hours, and that its worst value is 7 hours.

For KPI_3, as a binary indicator, the target is defined as 0 (no return) and the worst 1 (returned). Table 19 shows all the values corresponding to all three KPIs.

Table 19. The set of values of KPIs

KPI	*Worst value*	*Threshold value*	*Target value*
KPI_1	3 h	1.8 h	1 h
KPI_2	7h	5 h	3 h
KPI_3	1	-	0

After determining the details of the KPIs, in this section the tool *EnhancedLogMaker* was employed to enhance the sepsis event traces by adding three goal-related attributes. The enhanced sepsis log will be the main input of the tool *GoPED* to apply the algorithms on the log and select the cases based on different goal-related criteria. Then the CSV containing the IDs of the selected cases, together with the original sepsis event log, is fed to the tool *EventLogRefiner* to generate the filtered/refined event log. Such an event log was then used as the input for process discovery algorithms (i.e., IMF) to discover the model. The next section reports the goal-related criteria and the models mined based of them.

8.6. Process Models Resulting from the Use of GoPED

8.6.1 Algorithm 1 Applied to the Sepsis Log

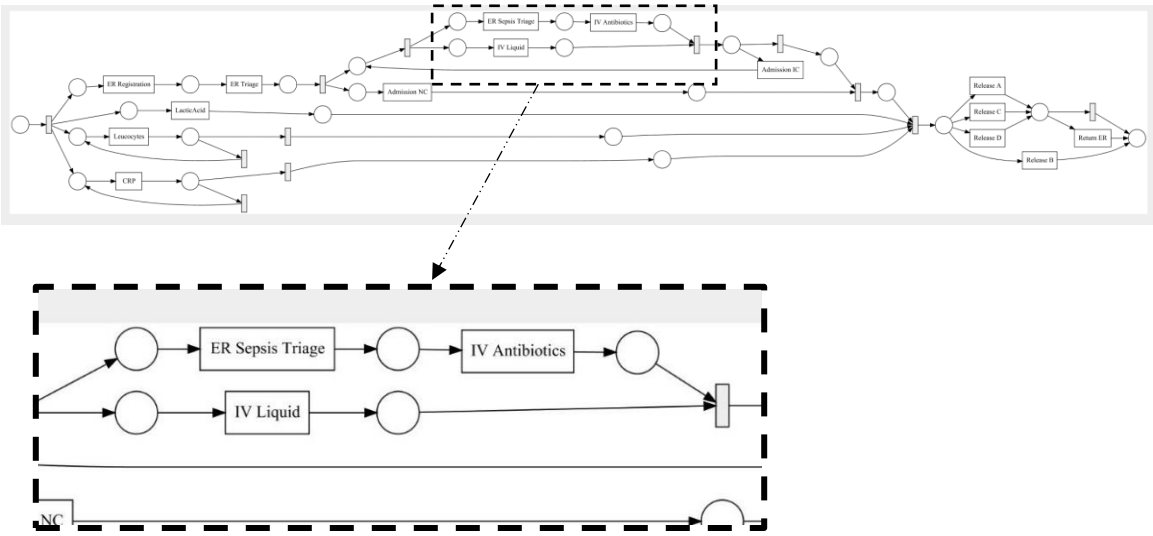
The first model mined from the enhanced sepsis log was from the case viewpoint, as considered in Algorithm 1 of the GoPED method. In this case, the goal-related criterion was as follows:

Goal criterion: S(G1) ≥ 95

Confidence Level = 0.9

The GoPED tools selected 328 cases with 320 unique traces, within 0.01 second. Model 5 is the model that ProM discovered using its IMF plugin.

Some difference between Model 5 and Model 4 (discovered from the original sepsis event log) can reveal some improvement. The weakness of the process in Model 4 that allows almost all activities be skipped and repeated is no longer a case in Model 5. The arc highlighted in Model 4 was the main path that allows repetition of all the activities before



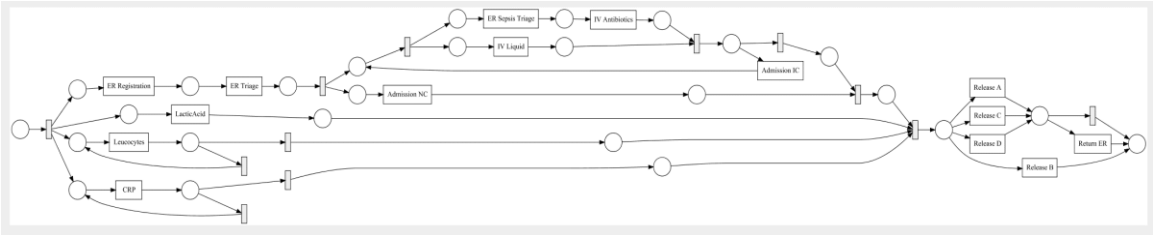
Model 5. Model generated to satisfy goal G1 for every single patient with a satisfaction level of at least 95 (with a confidence of 0.9)

Release A, and such an arc no longer exists in Model 5. Also, the silent activities that allowed skipping almost all activities are absent from Model 5. $Goal_1$ was related to administering antibiotics within one hour after sepsis triage. This goal was fully dissatisfied for 39% of the patients, and partially dissatisfied for 28% of the patients. Model 4 allows the activity *IV Antibiotics* not directly to follow *ER Sepsis Triage*. In that model, there are many different activities between *ER Sepsis Triage* and *IV Antibiotics*, and these two activities are not in a close temporal connection. In Model 5 however, the activity *IV Antibiotics* directly follows *ER Sepsis Triage*. This temporal relation is promising to help satisfy $Goal_1$, which is concerned with administrating antibiotics as soon as possible after the sepsis triage.

The second model mined from the enhanced sepsis log using Algorithm 1 was based on the following goal-related criteria:

Goal criteria: $S(G1) \geq 95$ $S(G2) \geq 90$

Confidence Level = 0.9



Model 6. Model generated to satisfy G_1 and G_2 for every single patient by satisfaction levels of at least 95 and 90 respectively (with a confidence of 0.9)

This time, in addition to $Goal_1$, $Goal_2$ was also considered to be satisfied by a value of at least 90 for all patients. The GoPED tool selected 317 cases with 309 unique traces within 0.013 seconds. Model 6, which was discovered by ProM using the IMF plugin, was identical to Model 5. This result is not surprising as the second considered goal could not have had a high impact on the selected cases. In fact, there are only 11 patients that have performed the lactic acid test longer than 3 hours after *ER Sepsis Triage*. 191 patients did not eventually perform the activity LacticAcid while they performed *ER Sepsis Triage*. About 95% of those patients either did not perform *IV Antibiotics* or did perform it within the recommended time.

8.6.2 Algorithm 2 Applied to the Sepsis Log

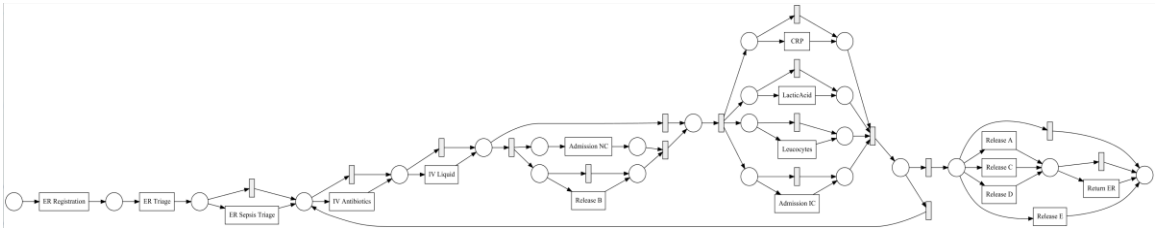
To apply the second algorithm of the GoPED method on the enhanced sepsis event log, the goal-related criteria (defined from a goal perspective) are as follows:

$$\text{Goal criteria: } s_{Agg.1} \geq 95 \quad \text{and} \quad s_{Agg.2} \geq 90$$

$$f1 = f2 = \text{Average}$$

According to Definition 2 and Definition 3 (in Section 5.1), the above criteria mean that the aggregated satisfaction levels of $Goal_1$ and $Goal_2$ should be at least 95 and 90, respectively. Also, the function used to calculate aggregations is the *Average* function. Here, GoPED is looking for the largest subset of cases whose average satisfaction in terms of the first two goals is equal or greater than the values defined in the criteria.

The GoPED tool selected 410 cases within 0.33 seconds. Model 7 is the model discovered by the IMF plugin of ProM.



Model 7. Model generated to keep the average satisfaction levels of G_1 and G_2 at least 95 and 90, respectively

Although Model 6 and Model 7 were both discovered based on goal-related criteria about $Goal_1$ and $Goal_2$, they are totally different from each other. The drawbacks of Model 1, discovered from the original event log, have returned. Model 7 is not exactly the same as Model 1, but it allows skipping and repeating almost all activities. The silent activities, parallel with almost all activities, showed up again and the arc that allows going back from the place before *Releases* to the place after *ER Sepsis Triage* became active again. However, the benefit of the immediate connection between the two activities *ER Sepsis Triage* and *IV Antibiotic* is preserved by Model 7.

Model 6 and Model 7 are both based on the satisfaction level of the same goals ($Goal_1$ and $Goal_2$). The threshold of the criteria for both models are also the same. However, the resulting models are much different. The only factor that is different between these two models is related to the viewpoint in which the criteria applied on the satisfaction levels. The criteria that result in Model 6 are from the case viewpoint (Algorithm 1) but the criteria that result in Model 7 considered the criteria from the goal viewpoint (Algorithm 2). From the case viewpoint, as explained in Section 4.1, every single case should meet the goal criteria individually (by a predefined confidence level). However, from the goal viewpoint, the aggregated satisfaction level (i.e., average of satisfaction levels) of a goal among all cases should meet the criteria.

In the sepsis context, Model 6 is based on the selected cases whose satisfaction levels for $Goal_1$ and $Goal_2$ are at least 95 and 90 respectively. But Model 6 is based on the selected cases whose *average* of satisfaction levels for $Goal_1$ and $Goal_2$ are at least 95 and 90 respectively. When the average is considered, some cases that individually do not meet the criteria can be selected by the algorithm. For example, some cases whose satisfaction

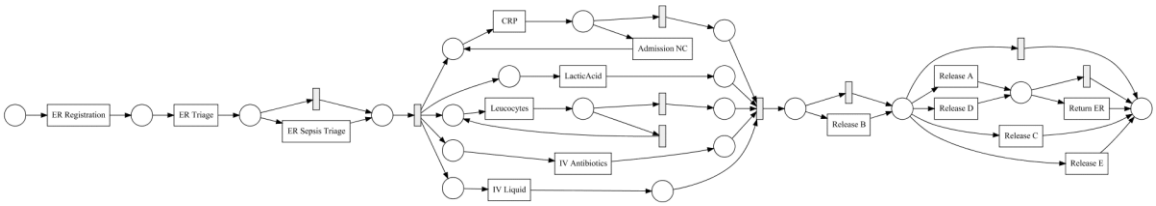
levels for $Goal_1$ and $Goal_2$ are less than 95 and 90 are selected for Model 7 because there are other cases whose satisfaction levels are close to or equal to 100 to compensate for such lower-satisfaction cases.

GoPED selected 317 cases for Model 6 and 410 cases for Model 7. It is not surprising that all 317 selected cases in Model 6 are also selected in Model 7. There are 93 cases selected for Model 7 that are not selected for Model 6. From these 93 cases, 84 cases do not meet the criteria $S(G1) \geq 95$ AND $S(G2) \geq 90$, but they are selected for Model 7 because some other cases with higher levels of satisfaction for the considered goals have compensated for these cases. There are also nine cases (out of those 93 cases) that individually meet the aforementioned criteria, but they have not been selected by Model 6 because of the *none-or-all* rule (i.e., they have shared traces with other cases that do not meet the criteria so that the confidence level criterion has not been met.)

The above analysis shows that in the sepsis case study, the goal viewpoint that uses the average as aggregation function may harm the resulting model. In that case study, the case viewpoint results in a model with fewer drawbacks.

8.6.3 Algorithm 3 Applied to the Sepsis Log

After applying the two first algorithms on the sepsis log, the third algorithm, which considers the goal-related criteria from the organization viewpoint, is applied in this section. In such a viewpoint, the aggregated satisfaction level of the entire goal model is used as one number known as the *comprehensive satisfaction level*. Algorithm 3 selected the largest subset of the cases that satisfy a criterion about the *comprehensive satisfaction level.* To this end, the goal model that defines the relation between the goals is required. Figure 24 shows the goal model assumed for the sepsis case study. In that goal model, $Goal_4$ is AND-decomposed by $Goal_1$ and $Goal_2$. Here, $Goal_4$ is "To comply with the medical guideline" and satisfied if and only if both sub-goals are satisfied. $Goal_3$ is promoting doing the medical treatment in a way such that the patient does not need to return to the hospital in the future. If the medical guideline is satisfied and the patient does not return to the hospital, we assume that the main goal "To perform the treatment properly" ($Goal_5$) is satisfied.



Model 8. The model when the comprehensive satisfaction level should be at least 95

Therefore, $Goal_5$ is AND-decomposed into $Goal_3$ and $Goal_4$. The simplified function that represents the goal model is defined as follows:

$$S(Goal_5) = Min\big(S(Goal_3), Min\big(S(Goal_1), S(Goal_2)\big)\big)$$

According to the goal model function, the aggregated satisfaction level of all three considered goals (connected to their KPIs) will determine the *comprehensive satisfaction level.*

The first model discovered for the enhanced sepsis log was based on the following goal-related criterion:

$$\text{Goal criterion: } S_{Comp} \geq 95$$

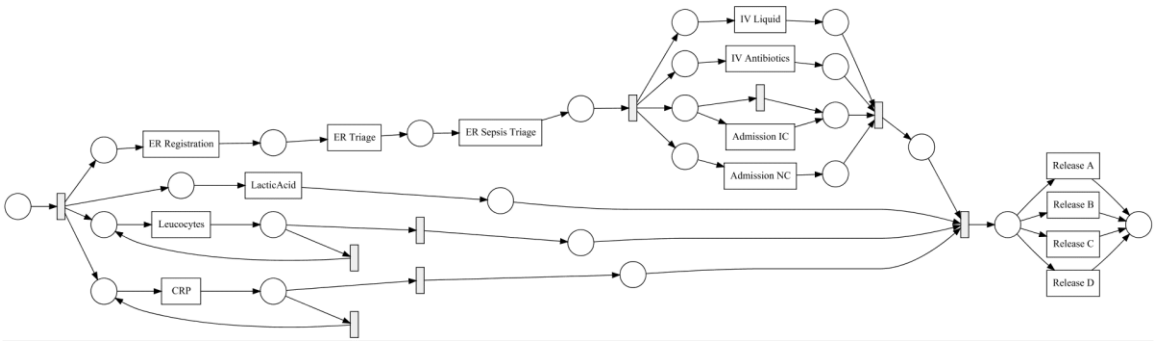
$$f_1 = f_2 = f_3 = \text{Average}$$

$$S_{Comp} = Min\big(S(Goal_3), Min\big(S(Goal_1), S(Goal_2)\big)\big)$$

The GoPED tool selected 295 cases within 0.35 seconds to satisfy the above criterion. Model 8 is the model discovered by the IMF plugin of ProM.

The drawback of skipping and repeating too many activities is solved in Model 8. Also, both activities *LacticAcid* and *IV Antibiotics* can be performed immediately after execution of *ER Sepsis Triage*. Such a structure is aligned with the goals "To decrease the time between ER Sepsis Triage and IV Antibiotics" and "To decrease the time between ER Sepsis Triage and LacticAcid".

Although Model 8 was discovered regarding a goal-related criterion that is satisfied only if the goal "To avoid returning to the hospital after discharge" is satisfied, it includes the activity *Return EI*. Existence of such an activity in Model 8 means that there are some cases selected by Algorithm 3 where the patient has returned to the hospital. According to the KPI values defined for $Goal_3$, this goal is either fully satisfied (i.e., satisfaction level is 100) or fully dissatisfied (i.e., satisfaction level is zero). Considering 295 cases selected by



Model 9. The model when the comprehensive satisfaction level should be 100

Algorithm 3, there are 14 cases who have returned to the hospital after discharge. The presence of the activity *Return EI* in Model 8 is resulting from the presence of those cases in the selected log. Similar to what was explained about Model 7, the function *average* for calculating the aggregation satisfaction level of $Goal_3$ is the main reason for having some cases with a satisfaction level of less than 95 (that means zero for $Goal_3$). Using *minimum* instead of Average as an aggregated function for $Goal_3$ would ensure that the selected traces would not include cases with returning patients.

The second model discovered using Algorithm 3 was obtained with a goal criterion requiring that the comprehensive satisfaction level be 100:

$$\text{Goal criterion: } S_{Comp}=100$$

$$f_1=f_2=f_3=\text{Average}$$

$$S_{Comp} = \text{Min}\big(S(Goal_3), \text{Min}\big(S(Goal_1), S(Goal_2)\big)\big)$$

The GoPED tool selected 208 cases within 0.15 seconds to satisfy the above criterion. Model 9 is the model discovered by the IMF plugin of ProM. The goal-related criterion required that the main goal of the goal model be fully satisfied. To fully satisfying the main goal of the Sepsis goal, all three considered goals should be fully satisfied. Therefore, Model 9 is based on the cases whose satisfaction levels of all three goals are 100. The average function does not have any impact here because to have an average of 100 (that is, the maximum number for a satisfaction level), all individual satisfactions must be 100. Therefore, Model 9 is discovered based on the patients whose lactic acid test has been done on time, whose antibiotics have been administered on time, and who have not returned to the hospital after discharge.

8.7. Chapter Summary

This chapter presented a case study where the GoPED method was applied to a real event log of a healthcare organization. The main characteristics of the event log were explained first, and then the table of traces of each patient (case) was generated using the *TraceMaker* tool, explained in Section 3.2.4. After consultation with a physician, the values of two KPIs driven from particular medical guidelines [184] were used to calculate satisfaction levels of their corresponding goals. The activity in the event log concerned with returning patients was used to calculate the third goal as a common goal of any healthcare system, which to minimize the number of patients who return to the hospital after they are discharged. A goal model (Figure 37) was made based on the three goals above. After determining KPIs, the tool *EnhancedLogMaker* was used to enhance the event traces by adding three goal-related attributes.

Then, the original event was fed to the *Alpha miner* and *Mine Petri net with Inductive Miner* (*IMF*) plugins in ProM 6.6. Then, two resulting models were analyzed, and some unignorable drawbacks were found in the models. Three algorithms of GoPED with different sets of goal-related criteria were applied to the enhanced log using the IMF plugin in ProM. The resulting models from different algorithms of GoPED were analyzed and compared to the model that was discovered from the whole original log. The drawbacks of the model mined from the original log were overcome using the GoPED method. Also, some insights about the process model were revealed by comparing and analyzing the deviations between the models resulting from GoPED and the original model. The models discovered using GoPED with different sets of goal-related criteria and different algorithms were also compared and analyzed. This comparison and analysis also revealed valuable insight about the process model and the ways of improving the model to be aligned with the goals.

Chapter 9. Related Work and Limitations

After explaining the GoPED method, assessing its algorithms, and using it on case study, in this chapter, the related papers reviewed in Section 2.6 and a few very recent papers will be considered again in comparison to GoPED. A comparison between the related work and this thesis will highlight how the proposed method fills several research gaps. Then, GoPED limitations and threats to the validity of this thesis are discussed.

9.1. Comparison with Related Work

In Section 2.5, we defined the queries to find the share of goal-oriented approaches in process mining studies. We aimed to determine whether goal-oriented process mining is a mature research field or whether there is currently a gap that exists. Then, we considered the contribution of 24 selected papers and categorized them into three main topics: Goal Modelling and Requirements Elicitation, Key Performance Indicators (KPIs), and Intention Mining. In Table 6, we showed the papers and their topic together with columns that described the goal modelling language that each paper has used, whether the paper has proposed a goal mining algorithm, the evaluation method used, whether the paper proposed a method that ends up in a process model, and whether the method aims to produce a model aligned with multiple goals.

Twelve out of these 24 selected papers proposed a goal mining algorithm. Goal mining is a scheme that considers finding the goals behind the activities performed by agents. In this context, it is assumed that when an agent decides to perform an activity or to violate a predefined process model, it is aware of the effects of the activity/violation and believes that it achieves its goal by performing such activity or violating the model [88] [89]. Thus, one can learn the agent's goal by methods of goal mining, some of which taking advantage of event logs. GoPED, however, does *not* aim to discover the intention or the goal behind the activities that are recorded in event logs. GoPED considers a higher level of abstraction (compared to the activity level) that is *trace*. Traces are the finest entity considered by GoPED to select the best subset of cases.

Table 20. Comparing selected papers related to goal-oriented process mining to GoPED

Category	*Paper*	*Goal model repre-sentation*	*Proposed goal mining algorithm?*	*Case study/ evaluation?*	*A process model as output?*	*Makes a model aligned with multiple goals?*
Goal Modelling & Requirements Elicitation	[87]	None	No	No	Yes	No
	[88]	None	Yes	Yes	No	No
	[89]	AND/OR graph	Yes	Yes	No	No
	[90]	AND/OR graph	Yes	Yes	No	No
	[91]	KAOS	No	Yes	Yes	No
	[92]	None	No	Yes	No	No
	[93]	None	No	No	Yes	No
	[94]	AND/OR graph	Yes	Yes	No	No
	[95]	None	No	Case study	No	No
	[96]	None	Yes	Yes	Yes	No
	This Thesis	**GRL**	**No, but filtering based on goals and KPIs**	**Yes**	**Yes**	**Yes**
KPI	[107]	Performance analysis	Performance analysis	Yes	No	No
	[108]	Performance analysis	Performance analysis	Yes	No	No
	[109]	Model redesigning with KPIs	Model redesigning with KPIs	Yes	Yes	No (one KPI)
	[110]	Performance analysis	Performance analysis	Yes	No	No
	[187]	No	No, but partitioning based on a KPI	Yes	Yes	No (one KPI)
	[188]	No	No, but filtering based on KPIs	Yes	Yes	Yes (only from cases' viewpoint)
Intention Mining	[97]	Mentioned and compared some of them		Yes	No	No
	[98]	MAP	Yes	No	No	No
	[99]	Mentioned and compared some of them		Yes	No	No
	[100]	MAP	Yes	No	No	No
	[101]	MAP	Yes	No	No	No
	[102]	MAP	Yes	No	No	No
	[103]	MAP	Compare	No	No	No
	[104]	MAP	Yes	No	No	No
	[105]	MAP	Yes	No	No	No
	[106]	MAP	Yes	No	No	No

In contrast to the papers that proposed a way to mine the goal behind the activities, GoPED aims to find a good sequence of activities that have finally resulted in satisfying pre-defined goals, measured with KPIs. Therefore, while Armentano and Amandi [88], Yan *et al.* [89] (both from the first category in Table 20) aim to find the goals from event logs, GoPED uses predefined goals as an input to refine the original event log and turn it into an event log to be used as a basis to discover a process model aligned with those predefined goals. Santiputri *et al.* [94] (from the first category in Table 20) aimed to mine the enterprise goal models from event logs. They mine goal refinement patterns from multi-layered event logs and compose goal refinement patterns to obtain *temporal correlation patterns* between goals and sub-goals and generate a goal model. GoPED, however, uses the goal model that is already defined to show the relation between goals.

The approach proposed by Horita *et al.* [90] is a goal-oriented conformance checking approach that can detect deviations between models and logs and analyze the effects of a deviation. Similarly, Outmazgin and Soffer [92] proposed a process mining-based analysis to study intentional incompliances, where agents intentionally deviate from the predefined procedures even if they are aware of them. Also, Dabrowski *et al.* [95] proposed an approach that leverages the synergy between *Process Mining, Requirements Engineering*, and *Crowdsourcing* for revising the existing system's functionalities and for ensuring that a software system operates as expected. Those three approaches are related to a Goal-oriented Conformance Checking (GoCC) approach identified as future work in the next chapter of this thesis.

Xu *et al.* [96] consider latent clinical topics as *clinical goals* and assume that the clinical activities performed in a specific time interval (such as hospitalization day) are prescribed for multiple clinical goals, and each clinical goal corresponds to a set of clinical activities. They categorize the detailed activities into some clinical goals, then apply a process discovery algorithm to the goal-based sequences (instead of the detailed clinical activities) to get a comprehensive process model [96]. This work is a type of clustering explained in Section 2.8 and does not aim to find a process model that satisfies a predefined goal as GoPED does. Baek *et al.* [87] considered the adaptability of processes discovered by process mining techniques in terms of being dynamically reconfigured for new requests made on business processes. The method proposed by Bernard and Andritsos [93] is a

clustering method that lets analysts define their own exploration goals for representing process models discovered from logs. Their method does not deal with agents or organization goals.

Ponnalagu *et al.* [91] consider different variants of an industry-scale business process that ideally contribute to achieving the organization's goals. They worked on the admission of a deviating process instance as a valid variant of the intended process that achieves the same goals as the intended process. Although they do not consider a predefined goal to make a new model, their idea that different variants of a trace can be accepted to satisfy a goal support the GoPED idea when different traces are selected regarding the same goal-related criterion.

The work of the category "Goal Modelling and Requirements Elicitation" shared with GoPED the idea of using goal models associated with process mining. The work of the category of Key Performance Indicators (KPIs) is mainly about using event logs to define/calculate the KPIs. GoPED, however, used KPIs as input for calculating the satisfaction levels of the predefined goals. The work of that category does not take advantage of KPIs to make process models aligned with goals. Dees *et al.* [109], however, proposed a methodology that takes event logs together with a process model (either discovered or designed manually) and repairs the process model with respect to the behaviours that do not violate any rule and have a significant improvement on a predefined KPI. This method can be used to make a process model aligned with one goal whose associated KPI is included in the event log. This approach is close to the first algorithm of GoPED where the goals are considered from the case viewpoint. However, the method considers one KPI as the performance of the process and does not deal with multiple ones.

The work categorized in the "Intention Mining" category is conducted mostly by the same research group. Intention mining shares ideas similar to process mining's, but here the goal is to discover *intentional process models* beyond activity process models. Intention mining is a kind of goal mining as it aims to work with intentions behind the activities.

Aligned with goal modelling characteristics, intentional process models specifically address the reasoning behind the activities. As discussed in Section 2.6.2, intention mining methods aim to get event logs to generate an intentional map. Such an approach is

different from GoPED where the main idea is to generate a process model aligned with goals.

In addition to the papers selected in the systematic literature review, Sedrakyan *et al.* [187] proposed a method that clusters the cases into multiple disjoint sets based on the current value of a specific KPI stored in the event log. Then the cases of each cluster are fed to the discovery algorithm, and the process model is discovered. Sedrakyan *et al.* [187] then compare the process models discovered from different clusters of cases and analyze the deviations between them. Such deviations with respect to the notion of the KPI can reveal some patterns that have impact on the value of the corresponding KPI. This method is close to Algorithm 1 of GoPED where the goals are considered from a case viewpoint. However, the none-or-all rule is not enforced in their method, and this can threaten the validity of the method when using real-world event logs.

Similar to the method proposed by Sedrakyan *et al.* [187], Gurgen Erdogan and Tarhan [188] used some KPIs to filter out some cases from the event log and discover the process model from the refined event log. To this end, they define goals and questions first, extract and preprocess the data, and then inspect the log to filter out undesirable cases and discover the process model. When analyzing the model, they answer questions, then they evaluate results and initiate proposals for process improvements. This proposed method is also a simplified version of Algorithm 1 of GoPED.

Both methods used in [187] and [188], are based on defining some goal-related criteria and checking every single case with such criteria and filtering it if it does not meet the criteria. The main part of the method is analyzing the difference between different process models they discover from different refined logs. In this sense, GoPED also provides different process models and analyzing their differences regarding the different goal-related criteria reveals some valuable insight about the process models.

9.2. Limitations and Challenges

There are several limitations to the current version of GoPED and challenges in its use. The challenges and limitations related to this research are rooted in the known challenges of both process mining and goal modelling.

9.2.1 Absence of Goal-related Information in Event Logs

As discussed in Section 2.4, the limitations related to *data* are the main challenges of process mining. In GoPED, an important new challenge is the absence of real-world logs that include some goal-related attributes (e.g., patient satisfaction) besides the usual event characteristics (e.g., timestamps). Although some potential KPIs can be extracted from existing logs (e.g., processing time), the goals of each trace and their satisfaction levels are seldom available and must be inferred from other sources.

9.2.2 Noise and Incompleteness in Selected Event Logs

As discussed in Section 4.14.1, an event log used in process discovery may not be a good representative for the process under consideration because of *noise* and *incompleteness*. The incompleteness of an event log, when discovering a model in the GoPED method, is a technical challenge that should be taken into consideration. Incompleteness can be a challenge when the selection of traces results in too few events to be used in process discovery. As an extreme example, assume that, using GoPED, we end up selecting very few cases based on a high level of satisfaction. Although the model discovered from a small log offers a high level of satisfaction, it becomes over-fitted and likely ungeneralizable for many cases. For that reason, GoPED is supposed to select as many traces as possible that satisfy the goal-oriented quality conditions. In these cases, one may consider a third research task for eliciting, through other means, activities that are poorly represented in event logs.

GoPED is likely robust when dealing with noisy sources because they can be filtered out while considering the traces' goals. However, for Algorithms 1 and 2, where the aggregation functions are used, the noise can affect the output in some cases. For example, such impact is observable when the average function is used to calculate the aggregated satisfaction level of a goal, as mathematically, the average function is not robust for dealing with noise. In contrast, the median and mode functions are more reliable for dealing with data that has noise. Therefore, some other functions such as mode and median deserve to be considered as aggregation functions for GoPED algorithms.

9.2.3 Relying on the Past to Improve the Future

Another challenge that might threaten the validity is about the term "guarantee" that we used in three goal-related criteria. The *none-or-all* rule for all three algorithms and the confidence level for Algorithm 1 were used to mitigate this concern. Similar to all knowledge discovery ideas, the main idea here is to learn from the past to configure a model that can result in a better future. The event logs selected by GoPED are aligned with goals, and the new models are inferred from such goal-aligned behaviours.

9.2.4 Concept Drift in the Requirements and Goals

In GoPED, we assume that the goal models and requirements will not change in unforeseen ways. For example, one can assume that the desired processes and the stakeholders' preferences that are extracted from logs do not change over time. Here, it is assumed that the past behaviour of the users is valid as a basis for designing a model for the future. Accordingly, *concept drift* [189], as an important phenomenon in the task of learning a model from data, should be considered. Goal-oriented Conformance Checking (GoCC), discussed in the future work (Section 10.2.4), might have some potential for finding some new requirements and/or goal model modifications based on the most recent behaviours to avoid using outdated goal models.

9.3. Threats to Validity

Runeson *et al*. [190] mention that "*The validity of a study denotes the trustworthiness of the results, and to what extent the results are not biased by the researchers' subjective point of view*". Section 2.7 already addressed threats to construct, internal, and external validity for the literature reviews. This section discusses potential threats to the validity (along the same three categories) of the thesis and its research, as well as some mitigation.

9.3.1 Construct Validity

Construct validity assesses the degree to which the used evaluation tools are able to answer the research question.

One threat is that only one major case study was used to evaluate GoPED in practice (Chapter 8), and it may not reflect the complexity of all real environments. This was partially mitigated by the careful selection of a public event log for which a paper and processes were available [185, 186], enabling some assessment and comparison.

One related threat is that this sepsis event log did not include explicit goals or goal models. A goal model had to be inferred from the informal description of the case study in the companion paper. This threat was partially mitigated by the input of a key informant (physician) who confirmed our understanding of potential indicators and their goals from the information found in the log and its associated documentation.

A third threat pertains to the use of synthetic event logs to assess the performance and scalability of GoPED in Chapter 7. Six different performance-related factors were explored in that experiment. Other might exist that would have an impact on the performance as well. In addition, each of these factors was experimented with in isolation, and combinations of two or more factors were not assessed. This was partially mitigated by selecting part of or growing an existing and real event log (from a children hospital) and by carefully covering size and distribution factors that are rather common in such experiments.

One last but important threat is that the usability of GoPED was not assessed by other users.

9.3.2 Internal Validity

Internal validity focuses on bias and other confounding factors. One such threat here is that bias might be introduced by having the thesis author select the event logs, perform the experiments, and analyze the results. This work might have been biased towards options for which GoPED's algorithms are favorable, for example, by using logs with large sets of attributes from which indicators and goal models can be inferred. This is partially mitigated by selecting existing event logs from real systems and by making the tools used in the thesis available online for others to replicate these results.

9.3.3 External Validity

The last category, external validity, focuses on the extent to which the evaluation results can be generalized to other situations or contexts. One important external threat in this thesis is that only two real event logs were used, and that they both come from the same domain (healthcare). Although there are no problems foreseen in using GoPED in other domains, there might be contexts or situations where different log structures or fewer goal-related attributes may not lead to results as positive as the ones observed here.

Chapter 10. Conclusion and Future Work

Goal-oriented process mining, as a novel research area bridging goal modelling and process mining, was introduced and elaborated in this thesis. The GoPED method, algorithms, and tools were also developed and assessed, and their potential for discovering process models aligned with predefined goals was evaluated through a performance assessment and a case study, with positive results. This last chapter provides a conclusion that recalls the contributions of the thesis and identifies important future work items to continue/complement the thesis' line of research.

10.1. Conclusion

This thesis focused on bridging the gap between process mining and goal modelling to achieve synergetic effects. The research question (RQ) was looking for a way to enrich conventional process mining activities to exploit both event logs and measurable goals to generate models aligned with goals. The research question was answered by proposing a new method, namely GoPED (Chapter 4), consisting of three algorithms to exploit the capabilities of goal modelling and perform process discovery in a goal-oriented fashion (Chapter 5). GoPED first enhances an event log by adding new goal-related information to all traces recorded in event logs. Then, it quantifies the satisfaction level of goals using a goal model that includes indicators. Such a goal model shows correlations between (often conflicting) goals of different stakeholders and allows what-if analysis and balancing trade-offs between confliction goals. Three types of goal-related criteria were introduced as the basis for generating goal-oriented models promising to achieve predefined goals. In order to discover models aligned with goals, the real behaviours that are aligned with the goals and achieve desired satisfaction levels are selected. The selected behaviours are the input of the process discovery algorithms to discover corresponding models. Three algorithms for such a selection were explicitly explained.

The main contribution of the thesis is the GoPED method (Chapter 4), which includes three algorithms for goal-oriented process mining (Chapter 5). A toolset containing

four tools to automate the GoPED method is a second contribution of this thesis. These tools are freely available for download at https://github.com/Mahdi-Ghasemi. The main *GoPED* tool (Chapter 7) automates the three algorithms of GoPED, whereas the three other tools enable preprocessing the input event log (*TraceMaker* and *EnhancedLogMaker*, see Section 4.2.4) and produce the final event log (*EventLogRefiner*, see Section 8.5) in a format ready to be consumed by existing process mining tools. Two extensive systematic literature reviews on process mining in general and goal-oriented process mining in particular, which are both published and well cited, also contribute to describing the state of the art in these areas (Chapter 2).

The feasibility and scalability of GoPED were assessed through 1960 different synthetic event logs with different sizes and configurations. The experiment showed that the GoPED tool could be used for the real-world sizes of event logs. Then, in a case study, the GoPED method was applied to a real log of a hospital in The Netherlands. The resulting models were compared to a model discovered from the original event log. New insights about the ability of different forms of process models to satisfy the goals were revealed. Learning from good behaviours that satisfy goals and detecting bad behaviours that hurt them is the opportunity that GoPED creates to redesign models better aligned with goals.

The results of three literature reviews reported in this thesis yield that potentially synergetic effects achievable by combining goal-oriented modelling and process mining have so far been neglected. Filling this gap can help practitioners of both domains deal with some of their own challenges difficult to mitigate separately.

Business analytics researchers and practitioner also can take advantage of GoPED capabilities in order to analyze and transform event logs into useful information, recognise and anticipate trends and their outcomes, and ultimately make data-driven business decisions. Discovering the underlying process models that are already followed in the organisation and adding the goal-related data that can represent the business value (through KPIs) enables researchers to consider the behaviours and trends associated with the goals and business values.

It is also our hope that the knowledge about both domains presented in this thesis will help researchers consider the potential synergy between goal modelling and process mining. Moreover, this thesis is valuable to requirements engineers interested in obtaining

evidence-based definitions of existing processes, their goals, and indicators. This enables them to measure goal satisfaction to assess alignment between goals and processes. Practitioners can further benefit from this thesis as it raises awareness about the limitations of current activity-based process modelling and mining approaches and about emerging GoPED techniques that attempt to discover goal-aligned process models.

Requirements engineers who practice scenario-based elicitation methods can also exploit GoPED capabilities to generalize instance-level event logs into models and analyze different models discovered through different goal criteria. This allows them to take advantage of such data, which is orders of magnitude larger than the interview or feedback data that requirements engineers often use.

The emerging research directions enabled by the availability of GoPED could lead to developments that would help practitioners understand and improve processes, manage requirements, and support organizations in better measuring and achieving their goals.

10.2. Future Work

Some of the limitations and threats identified in the previous chapter can be addressed by several future work items.

10.2.1 Improvements to the GoPED Algorithms and Tool

For GoPED algorithms, it was assumed, for simplicity, that the aggregation function (explained as Function f_j in Definition 2) used to find aggregated satisfaction level of a goal among all cases is *average*. There might be some situations where the modeller needs to use some different aggregation functions. Maximum, Minimum, Median, and Mode are examples of such functions. The GoPED algorithms can be improved by getting the aggregation function as an input. Different aggregation functions will also require different formations of the optimization problem used in Algorithms 2 and 3.

It was assumed in GoPED's algorithms that the goal-related criteria are generated through AND composition between multiple criteria. Definition 3 described Q_{case} as the logical conjunction (AND) of a set of criteria q_j. For example, for Algorithm 2, the goal criteria of {G1≥90%, G3≥80%} means G1≥90% AND G3≥80%. In contrast, a modeller

might be interested in finding a subset of cases that satisfy G1 by at least 90% *OR* G3 by at least 80%. In such a situation, the formation of the optimization problems of Algorithm 1 and Algorithm 2 will change. However, improving these algorithms to accommodate different logical operators between goal-related criteria more simply requires further implementation effort.

10.2.2 Using the Ratio of Selected Cases to Total Cases as a Metric

The main output of the GoPED algorithms is the subset of cases that are selected regarding the goal-related criteria. It is noteworthy that the goal-related criteria should not necessarily determine minimum satisfaction levels to find the model of *good* behaviours. The goal-related criteria can also define maximum satisfaction levels. When GoPED is looking for a model that is poorly aligned with goals, a maximum limit can be defined. In this case, a model of *bad* behaviours that violate the goals will be discovered. Such a model can be used to find some insight about the model to help the analyst improve the model. Also, it can be used to define the patterns that should be avoided in the organization.

In both cases, it will be beneficial to consider what *proportion* of total cases were selected by GoPED. The proportion can be considered as a metric that shows how aligned with goals the current process is. When GoPED is looking for models of good behaviours (the goal-related criteria are forcing minimum thresholds for satisfaction levels), if such metric is low, then the current underlying process is poorly aligned with goals. If that metric is too low, the process model discovered from GoPED will be suffering from "incompleteness" and be extremely overfitted. In this case, although the resulting process model is not generalized enough to be used as an alternative for the underlying process model, it can still provide valuable insights about how to improve the process. On the other hand, if such metric is high enough (e.g., close to 1), the analyst can conclude that the current process is in good shape and its alignment with goals is satisfactory.

The analyst can change the threshold of goal-related criteria and monitor the trend of such a metric. Such a trend could be analyzed to reveal insight into the current process. How to do this pragmatically remains a research question.

10.2.3 Regrouping All GoPED Tools as a Plugin for Process Mining Tools

As shown in Figure 22, this thesis provided four tools used in the GoPED method to generate a new event log from an original one according to some goal-related criteria. All the tools are currently used one by one, with some CSV files for event logs and traces, and with a goal model and goal criteria as additional inputs. Integrating these four tools in one single tool and making it available as a plugin for ProM and other process mining environments would be a practical step to increase the visibility of GoPED.

10.2.4 Improving the Validation

There is a need for the GoPED method to be used on event logs from different domains (to improve the generalization of the results) by different practitioners and researchers (to improve usability and reduce bias in its assessment). This would also provide a variety of contexts where the presence or absence of goal-related information could further be studied.

10.2.5 Combining GoPED with Other Filtering and Clustering Approaches

Although the thesis focused on GoPED, in practice, GoPED will unlikely be sufficient as the only filtering method to be applied on event logs. Basic filtering approaches such as removing infrequent events or truncated cases, or filtering out events/cases based on simple attributes, only to name a few, likely remain important tools in most situations. Similarly, trace clustering is also something that can help reduce the complexity of event logs without loosing important information (see Section 2.8). How best to combine filtering, clustering, and GoPED for preprocessing event logs efficiently, effectively, and in an automated way remains a research topic.

10.2.6 Towards Goal-oriented Conformance Checking (GoCC)

The elicitation of requirements from huge collections of data is becoming a research trend aligned with increasing research attention dedicated to big data [191]. Data-driven and user-centered approaches [125] aligned with *crowd-based requirements engineering* (CrowdRE) [123] are automated or semi-automated approaches that aim to collect and analyze data from a crowd to elicit validated user requirements [125], [192].

In particular, process discovery techniques use a form of crowdsourcing where event logs are generated from system users executing tasks. This can help requirements engineers generate *as-is* process models that reveal real behaviours recorded in logs. In this sense, process dicovery can be employed as a data-driven substitute or complement to use case or scenario elicitation. In scenario-based requirements engineering [193], scenarios represent paths of possible behaviours through a use case, and they are considered to elaborate requirements. In fact, scenario-based RE and process mining both consider ways of generalizing instance-level data into models, except that in a process mining context such data is orders of magnitude larger. Scenario-based techniques (e.g., [194], [195], [130]), however, do not scale to process mining-size problems and the ability of process mining to deal with huge data can offer RE a solution for this challenge.

One research topic that can complement the GoPED method and further enable it to be used in other Requirements Engineering research work is *Goal-oriented Conformance Checking* (GoCC). Conformance checking, the second type of process mining activity, is mainly concerned with comparing what *has* happened with what *should have* happened. Here, an existing or *prescribed* process model is compared to an event log or a discovered model. Such a comparison highlights where the real process deviates from the prescribed process. Conformance checking allows quantifying the level of conformance and diagnosing differences.

While GoPED aims to discover goal-aligned process models, GoCC, shown in Figure 38, does not deal with process discovery algorithms. Instead, it waits for the process model discovered through mining as a representation of real-world behaviours. Then, it takes this real process as an input together with the prescribed process model. The new input here is the goal model. The real and prescribed processes are compared to find and analyze misalignments. Such an analysis enables updating the goal models on the one hand, and to revise the prescribed (designed) process model on the other hand.

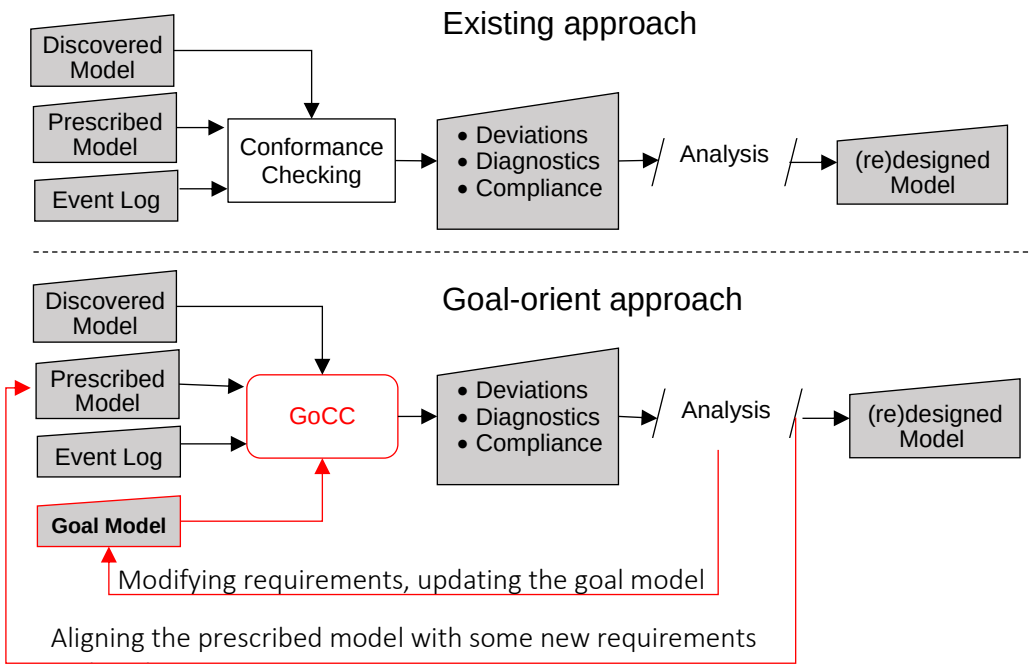


Figure 38. Goal-oriented conformance checking (GoCC)

Different metrics and algorithms are developed to check the conformance of models to models, logs to models, and logs to logs and find misalignments and degrees of deviation. There exist some concepts introduced in process mining dealing with stakeholders' requirements. *Auditing* is one potential usage of conformance checking to check whether business processes are performed within the boundaries set by stakeholders [196].

Conformance checking is also a means to reveal where the real process has deviated from a prescribed process model. In case of exposing undesirable deviations, the need for better control of the process is implied. However, GoCC also leverages desirable deviations to reconsider the requirements and update the goal models by taking into account the deviations that stakeholders accept.

As a mock example, shown in Figure 39, the footsteps of students walking on campus can be considered as an event log of the process of "*going from building A to building B*". The footsteps logged on the lawn (the more frequent the path, the deeper the footsteps) can represent the event log. Path *P* is considered as the model that is discovered from the log. Such a model is not necessarily the same as the prescribed model that may have been previously designed (path *Q*). In this example, there exists a prescribed model, but there

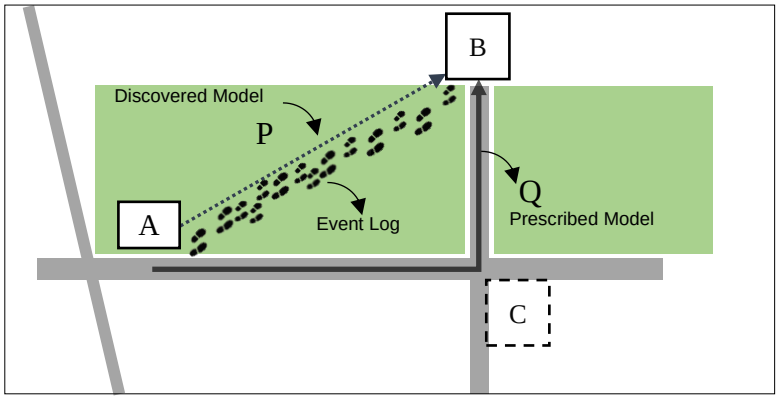


Figure 39. Footsteps of students against the sidewalks

may also exist many cases where a priori models are not available. In such cases, process discovery can generate a model that can serve at least as a valuable start point for eliciting stakeholders' approaches towards their goals.

When there exists a prescribed model, differences between that model and the discovered one or the logs (e.g., activities in the model not existing in the log or vice versa) can be further analyzed. These deviations can serve as a source for eliciting new requirements or validation and/or modification of the current ones. Accordingly, requirements engineers can use *conformance checking* capabilities to find new system/process requirements to improve the alignment between real practices and desired ones. Frequent deviations in an event log should trigger the requirements engineer to re-assess the suitability of the system's goals, requirements, and processes. In the example shown in Figure 39, the footsteps of students (event log) may not be aligned with the constructed sidewalks (prescribed model). These footsteps may show that the desired way is the shortest path between points *A* and *B*. The goal behind the students here is "*to increase the speed*" or "*to decrease process time*", i.e., some goal (or requirement) that has not been included in the original goal model or documented requirements. As a result, the modeller can update the requirements or/and goal model and modify the prescribed process model to accommodate such a deviation. Conversely, a goal or a requirement that is entirely satisfied could be of more interest for the future.

Let us assume that the students will frequently use the sidewalk (prescribed model $A \rightarrow C \rightarrow B$) on snowing days. At this time there are very rare footsteps on the path *P* (A→B).

Students have changed their behaviour as their goals' or requirements' priorities are changed. In this case, the goal behind the students' behaviour is "*to increase safety (from sliding)*" or "*to decrease the risk of making shoes muddy*". Nevertheless, the requirements "*to increase the speed*" *and* "*to decrease process time*" are yet pursued but their priorities are lower than the safety-related requirements or goals raising in snowy conditions. Frequent deviations and/or conformances in different situations highlight some knowledge of importance about different requirements and goals.

Hence, when a deviation in an event log is considerably frequent, GoCC, as a form of requirements engineering technique that exploits *crowdsourcing*, could find a new requirement (or new goal). When the purpose is to (re)design a process model, GoCC considers event data specifically to find deviations and to address how actual participants in the process-to-be performed it themselves. This data-driven method will find some requirements through the logs, instead of eliciting them from stakeholder goals or situations documented in the goal models.

References

[1] W. van der Aalst, Process Mining Data Science in Action, 2 ed., Springer-Verlag Berlin Heidelberg, 2016.

[2] D. Amyot and G. Mussbacher, "User Requirements Notation: The First Ten Years, The Next Ten Years," *Journal of Software,* vol. 6, no. 5, pp. 747-768, 2011.

[3] M. Ghasemi and D. Amyot, "From Event Logs to Goals: A Literature Review of Goal-oriented Process Mining," *Requirements Engineering,* 2018. Submitted.

[4] R. H. Von Alan, S. T. March, J. Park and S. Ram, "Design Science in Information Systems Research," *MIS Quarterly,* vol. 28, no. 1, pp. 75-105, 2004.

[5] K. Peffers, T. Tuunanen, M. A. Rothenberger and S. Chatterjee, "A Design Science Research Methodology for Information Systems Research," *Journal of Management Information Systems,* vol. 24, no. 3, pp. 45-77, 2007.

[6] M. Ghasemi and D. Amyot, "Process mining in healthcare: a systematised Literature Review," *International Journal of Electronic Healthcare,* vol. 9, no. 1, pp. 60-88, 2016.

[7] M. Ghasemi, "Towards Goal-oriented Process Mining," in *Requirements Engineering Conference (RE), 2018 IEEE 26th International*, IEEE CS, 2018, pp. 484-489.

[8] M. Ghasemi, "What requirements engineering can learn from process mining," in *2018 1st International Workshop on Learning from other Disciplines for Requirements Engineering (D4RE)*, IEEE, 2018, pp. 8-11.

[9] M. Ghasemi and D. Amyot, "Data preprocessing for goal-oriented process discovery," in *IEEE 27th International Requirements Engineering Conference Workshops (REW)*, 2019.

[10] M. Ghasemi and D. Amyot, "Goal-oriented Process Enhancement and Discovery," in *International conference on business process management.*, 2019.

[11] D. Amyot, O. Akhigbe, M. Baslyman, S. Ghanavati, M. Ghasemi, J. Hassine, L. Lessard, G. Mussbacher, K. Shen and E. Yu, "Combining Goal modelling with Business Process modelling – Two Decades of Experience with the User Requirements Notation Standard," *Enterprise Modelling and Information Systems Architectures (EMISAJ),* 2021.

[12] B. Kitchenham, "Procedures for performing systematic reviews," *Keele, UK, Keele University,* 2004.

[13] A. Rozinat and W. van der Aalst, "Conformance checking of processes based on monitoring real behavior," *Information Systems,* vol. 33, no. 1, pp. 64-95, 2008.

[14] W. van der Aalst, Process mining: discovery, conformance and enhancement of business processes, Springer Science & Business Media, 2011.

[15] L. Aldin and S. de Cesare, "A literature review on business process modelling: new frontiers of reusability," *Enterprise Information Systems,* vol. 5, no. 3, pp. 359-383, aug 2011.

[16] J. E. Cook, "Process discovery and validation through event-data analysis," 1996.

[17] R. Agrawal, D. Gunopulos and F. Leymann, "Mining process models from workflow logs," in *International Conference on Extending Database Technology*, 1998.

[18] W. van der Aalst, T. Weijters and L. Maruster, "Workflow mining: Discovering process models from event logs," *IEEE Transactions on Knowledge and Data Engineering,* vol. 16, no. 9, pp. 1128-1142, 2004.

[19] W. van der Aalst and C. Stahl, Modeling Business Processes: A Petri Net-Oriented Approach, The MIT Press, 2011.

[20] C. Petri and W. Reisig, "Petri net," 2008. [Online]. Available: http://www.scholarpedia.org/article/Petri_net.

[21] OMG (Object Management Group), "Business Process Model and Notation, Version 2.0," 2011.

[22] W. van der Aalst, "What makes a good process model?," *Software & Systems Modeling*, vol. 11, no. 4, pp. 557-569, 2012.

[23] J. Wang, R. K. Wong, J. Ding, Q. Guo and L. Wen, "Efficient Selection of Process Mining Algorithms," *IEEE Transactions on Services Computing*, vol. 6, no. 4, pp. 484-496, 2013.

[24] A. Weijters and J. J. Ribeiro, "Flexible Heuristics Miner (FHM)," in *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, 2011.

[25] A. K. A. de Medeiros, A. J. M. M. Weijters and W. van der Aalst, "Genetic process mining: an experimental evaluation," *Data Mining and Knowledge Discovery*, vol. 14, no. 2, pp. 245-304, jan 2007.

[26] R. Bergenthum, J. Desel, R. Lorenz and S. Mauser, "Process mining based on regions of languages," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2007.

[27] A. Rozinat, A. D. Medeiros and C. Günther, Towards an evaluation framework for process mining algorithms, 2007.

[28] B. van Dongen, A. K. Alves de Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters and W. van der Aalst, "The ProM framework: A new era in process mining tool support," *Application and Theory of Petri Nets 2005*, no. 3536, pp. 444-454, 2005.

[29] "ProM Tools," Process Mining Group, Eindhoven Technical University, [Online]. Available: http://www.promtools.org/. [Accessed 2018].

[30] A. de Medeiros, W. van der Aalst and C. Pedrinaci, "Semantic process mining tools: core building blocks," 2008.

[31] IEEE, "IEEE Standard for eXtensible Event Stream (XES)," IEEE , 2016. [Online]. Available: http://www.xes-standard.org/. [Accessed 2018].

[32] J. Claes and G. Poels, "Process mining and the ProM framework: an exploratory survey," in *Business Process Management Workshops*, 2013.

[33] "Disco," [Online]. Available: https://fluxicon.com/disco/.

[34] Software AG, "ARIS Process Performance Manager," [Online]. Available: https://www.softwareag.com/corporate/products/aris_alfabet/bpa/aris_ppm/default. [Accessed 2018].

[35] "Celonis," [Online]. Available: https://www.celonis.com/intelligent-business-cloud/process-discovery/. [Accessed 2018].

[36] QPR Software Plc, "QPR ProcessAnalyzer," [Online]. Available: http://www.qpr.com/products/qpr-processanalyzer. [Accessed 2018].

[37] ICRIS, "Process Mining Factory," 2018. [Online]. Available: http://www.processminingfactory.nl.

[38] W. van der Aalst, A. Adriansyah, A. K. A. De Medeiros, F. Arcieri, T. Baier, T. Blickle, J. C. Bose, P. van den Brand, R. Brandtjen and J. Buijs, "Process mining manifesto," in *International Conference on Business Process Management*, Springer, 2011, pp. 169-194.

[39] R. Mans, W. van der Aalst and R. J. Vanwersch, Process Mining in Healthcare: Evaluating and Exploiting Operational Healthcare Processes, Springer, 2015.

[40] D. Amyot, J. Horkoff, D. Gross and G. Mussbacher, "A lightweight GRL profile for i* modeling," *International Conference on Conceptual Modeling*, pp. 254-264, 2009.

[41] A. van Lamsweerde, "Goal-oriented requirements engineering: A guided tour," in *Proceedings of the IEEE International Conference on Requirements Engineering*, IEEE, 2001, pp. 249-261.

[42] A. Cailliau and A. van Lamsweerde, "Integrating exception handling in goal models," *22nd International Requirements Engineering Conference (RE)*, pp. 43-52, 2014.

[43] A. van Lamsweerde, R. Darimont and E. Letier, "Managing Conflicts in Goal-Directed Requirements Engineering," *IEEE Transactions on Software Engineering*, vol. 24, no. 11, pp. 908-925, 1998.

[44] J. Hassine and D. Amyot, "An empirical approach toward the resolution of conflicts in goal-oriented models," *Software & Systems Modeling*, vol. 16, no. 1, pp. 279-306, 2017.

[45] A. van Lamsweerde, "Goal-oriented requirements engineering: a roundtrip from research to practice," *Proceedings. 12th IEEE International Requirements Engineering Conference, 2004.*, pp. 4-7, 2004.

[46] A. van Lamsweerde and E. Letier, "Handling obstacles in goal-oriented requirements engineering," *IEEE Transactions on Software Engineering*, vol. 26, no. 10, pp. 978-1005, 2000.

[47] J. Horkoff, T. Li, F.-L. Li, M. Salnitri, E. Cardoso, P. Giorgini and J. Mylopoulos, "Using goal models downstream: a systematic roadmap and literature review," *International Journal of Information System Modeling and Design (IJISMD)*, vol. 6, no. 2, pp. 1-42, 2015.

[48] A. van Lamsweerde, "Requirements engineering: from craft to discipline," *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*, pp. 238-249, 2008.

[49] C. Rolland, N. Prakash and A. Benjamen, "A Multi-Model View of Process Modelling," *Requirements Engineering*, vol. 4, no. 4, pp. 169-187, dec 1999.

[50] L. Chung, B. A. Nixon, E. Yu and J. Mylopoulos, Non-functional requirements in software engineering, vol. 5, Springer Science & Business Media, 2012.

[51] E. Yu, "Modelling strategic relationships for process reengineering," 1995.

[52] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia and J. Mylopoulos, "Tropos: An Agent-Oriented Software Development Methodology," *Autonomous Agents and Multi-Agent Systems*, vol. 8, no. 3, pp. 203-236, may 2004.

[53] ITU-T, "Recommendation Z.151 (10/18) User Requirements Notation (URN) – Language definition," 2018. [Online]. Available: https://www.itu.int/rec/T-REC-Z.151-201810-I/en.

[54] A. Pourshahid, D. Amyot, L. Peyton, S. Ghanavati, P. Chen, M. Weiss and A. J. Forster, "Business process management with the user requirements notation," *Electronic Commerce Research*, vol. 9, no. 4, pp. 269-316, 2009.

[55] D. Amyot, S. Ghanavati, J. Horkoff, G. Mussbacher, L. Peyton and E. Yu, "Evaluating goal models within the goal-oriented requirement language," *International Journal of Intelligent Systems*, vol. 25, no. 8, pp. 841-877, 2010.

[56] A. Anda and D. Amyot, "An Optimization Modeling Method for Adaptive Systems Based on Goal and Feature Models," in *Tenth International Model-Driven Requirements Engineering (MoDRE)*, 2020.

[57] C. Rolland and C. Salinesi, "Modeling goals and reasoning with them," in *Engineering and Managing Software Requirements*, Springer, 2005, pp. 189-217.

[58] H. K. Kraiem, J. Dimassi and Z. Al Khanjari, "Mapping from MAP models to BPMN processes," *Journal of Software Engineering*, vol. 8, no. 4, pp. 252-264.

[59] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey and S. Linkman, *Systematic literature reviews in software engineering - A systematic literature review*, vol. 51, 2009, pp. 7-15.

[60] N. Agarwal and L. Singh, "Process mining tools: A comparative analysis and review," *Advances in Computer Science and Information Technology (ACSIT)*, vol. 1, no. 2, pp. 26-29, 2014.

[61] A. Tiwari, C. J. Turner and B. Majeed, "A review of business process mining: state-of-the-art and future trends," *Business Process Management Journal*, vol. 14, no. 1, pp. 5-22, 2008.

[62] A. R. C. Maita, L. C. Martins, C. R. Lopez Paz, S. M. Peres and M. Fantinato, "Process mining through artificial neural networks and support vector machines," *Business Process Management Journal*, vol. 21, no. 6, pp. 1391-1415, nov 2015.

[63] D. Breuker and M. Matzner, "Performances of business processes and organizational routines: Similar research problems, different research methods - A literature review," in *ECIS 2014 Proceedings - 22nd European Conference on Information Systems*, 2014.

[64] D. Yue, X. Wu, H. Wang and J. Bai, "A review of process mining algorithms," in *usiness Management and Electronic Information (BMEI), 2011 International Conference on*, 2011.

[65] Z. Qing-tian, "A Survey of Research Issues and Approaches on Process Mining," *Journal of System Simulation,* vol. 19, pp. 275-280, 2007.

[66] J. Wang, D. Tang and Y. Xie, "A Survey of Research on Process Mining Algorithms," *Fire Control & Command Control,* vol. 36, no. 8, pp. 5-10, 2011.

[67] W. Yang and Q. Su, "Process mining for clinical pathway: Literature review and future directions," in *11th International Conference on Service Systems and Service Management (ICSSSM)*, 2014.

[68] E. Rojas, M. Arias and M. Sepúlveda, "Clinical processes and its data, what can we do with them?," in *HEALTHINF 2015 - 8th International Conference on Health Informatics, Proceedings; Part of 8th International Joint Conference on Biomedical Engineering Systems and Technologies, BIOSTEC 2015*, 2015.

[69] E. Rojas, J. Munoz-Gama, M. Sepúlveda and D. Capurro, "Process mining in healthcare: A literature review," *Journal of biomedical informatics*, vol. 61, pp. 224-236, 2016.

[70] A. Medeiros, A. Weijters and W. van der Aalst, "Using genetic algorithms to mine process models: representation, operators and results," *Event London,* no. i, pp. 1-38, 2005.

[71] W. van der Aalst and M. Song, "Mining Social Networks: Uncovering Interaction Patterns in Business Processes," *Proceedings of the 2nd International Conference on Business Process Management,* pp. 244-260, 2004.

[72] J. E. Cook and A. L. Wolf, "Discovering models of software processes from event-based data," *ACM Transactions on Software Engineering and Methodology,* vol. 7, no. 3, pp. 215-249, 1998.

[73] G. Schimm, "Mining exact models of concurrent workflows," *Computers in Industry,* vol. 53, no. 3, pp. 265-281, 2004.

[74] I. Ailenei, A. Rozinat, A. Eckert and W. van der Aalst, "Definition and validation of process mining use cases," *Lecture Notes in Business Information Processing,* vol. 99 LNBIP, no. PART 1, pp. 75-86, 2012.

[75] R. Jagadeesh Chandra Bose, R. Mans and W. van der Aalst, "Wanna Improve Process Mining Results ? It ' s High Time We Consider Data Quality Issues Seriously," *Proceedings of the 2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM),* pp. 127-134, 2013.

[76] C. Günther and W. van der Aalst, "Fuzzy Mining – Adaptive Process Simplification Based on Multi-perspective Metrics," *Business Process Management - Lecture Notes in Computer Science,* vol. 4714, pp. 328-343, 2007.

[77] A. J. M. M. Weijters, W. van der Aalst and A. A. De Medeiros, "Process Mining with the HeuristicsMiner Algorithm," *Cirp Annals-manufacturing Technology,* vol. 166, pp. 1-34, 2006.

[78] M. Song and W. van der Aalst, "Supporting process mining by showing events at a glance," *In Proceedings of 17th Annual Workshop on Information Technologies and Systems,* pp. 139-147, 2007.

[79] J. Li, D. Liu and B. Yang, "Process Mining: Extending α-Algorithm to Mine Duplicate Tasks in Process Logs," *Advances in Web and Network Technologies, and Information Management,* pp. 396-407, 2007.

[80] F. Caron, J. Vanthienen, K. Vanhaecht, E. Van Limbergen, J. Deweerdt and B. Baesens, "A process mining-based investigation of adverse events in care processes.," *Health Information Management Journal,* vol. 43, no. 1, pp. 16-25, oct 2013.

[81] L. De Bleser, R. Depreitere, K. De Waele, K. Vanhaecht, J. Vlayen and W. Sermeus, "Defining pathways," *Journal of Nursing Management,* vol. 14, no. 7, pp. 553-563, 2006.

[82] Process Mining Group, "Papers About Process Mining in Healthcare," Eindhoven University of Technology, 2014. [Online]. Available: http://www.processmining.org/health/papers.

[83] R. S. Mans, W. van der Aalst, R. J. Vanwersch and A. J. Moleman, "Process Mining in Healthcare: Data Challenges When Answering Frequently Posed Questions," *Process Support and Knowledge Representation in Health Care,* pp. 140-153, 2013.

[84] M. Song, C. W. Günther and W. Van der Aalst, "Trace clustering in process mining," in *International Conference on Business Process Management*, Springer, 2008, pp. 109-120.

[85] Á. Rebuge and D. R. Ferreira, "Business process analysis in healthcare environments: A methodology based on process mining," *Information Systems,* vol. 37, no. 2, pp. 99-116, apr 2012.

[86] C. Fernández-Llatas, J.-M. Benedi, J. García-Gómez and V. Traver, "Process Mining for Individualized Behavior Modeling Using Wireless Tracking in Nursing Homes," *Sensors,* vol. 13, no. 11, pp. 15434-15451, nov 2013.

[87] S.-J. Baek, J.-W. Ko, G.-J. Kim, J.-S. Han and Y.-J. Song, "Goal-Heuristic Analysis Method for an Adaptive Process Mining," *Proceedings of the International Conference on IT Convergence and Security 2011,* pp. 409-418, 2012.

[88] M. G. Armentano and A. A. Amandi, Towards a goal recognition model for the organizational memory, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 730-742.

[89] J. Yan, D. Hu, S. S. Liao and H. Wang, "Mining Agents' Goals in Agent-Oriented Business Processes," *ACM Transactions on Management Information Systems,* vol. 5, no. 4, pp. 1-22, 2014.

[90] H. Horita, H. Hirayama, Y. Tahara and A. Ohsuga, "Towards Goal-Oriented Conformance Checking," 2015.

[91] K. Ponnalagu, A. Ghose, N. C. Narendra and H. K. Dam, Goal-aligned categorization of instance variants in knowledge-intensive processes, vol. 9253, H. R. Motahari-Nezhad, J. Recker and M. Weidlich, Eds., Cham: Springer International Publishing, 2015, pp. 350-364.

[92] N. Outmazgin and P. Soffer, "A process mining-based analysis of business process work-arounds," *Software & Systems Modeling,* vol. 15, no. 2, pp. 309-323, may 2016.

[93] B. G. Bernard and P. Andritsos, "CJM-ex: goal-oriented exploration of customer journey maps using event logs and data analytics," in *15th International Conference on Business Process Management (BPM 2017)*, 2017.

[94] M. Santiputri, N. Deb, M. A. Khan, A. Ghose, H. Dam and N. Chaki, "Mining Goal Refinement Patterns: Distilling Know-How from Data," 2017, pp. 69-76.

[95] J. Dabrowski, F. M. Kifetew, D. Munante, E. Letier, A. Siena, A. Susi, J. Dabrowski, F. M. Kifetew, D. Munante, E. Letier, A. Siena and A. Susi, "Discovering Requirements through Goal-Driven Process Mining," in *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*, 2017.

[96] X. Xu, T. Jin, Z. Wei and J. Wang, "Incorporating Topic Assignment Constraint and Topic Correlation Limitation into Clinical Goal Discovering for Clinical Pathway Mining," *Journal of Healthcare Engineering,* vol. 2017, 2017.

[97] G. Khodabandelou, "Contextual recommendations using intention mining on process traces," *Doctoral Consortium, Seventh International Conference on Research Challenges in Information Science,* 2013.

[98] G. Khodabandelou, C. Hug, R. Deneckere and C. Salinesi, "Supervised intentional process models discovery using Hidden Markov models," in *2013 IEEE Seventh International Conference on Research Challenges in Information Science (RCIS)*, IEEE, 2013, pp. 1-11.

[99] G. Khodabandelou, C. Hug, R. Deneckere and C. Salinesi, "Process mining versus intention mining," *Lecture Notes in Business Information Processing,* vol. 147, pp. 466-480, 2013.

[100] G. Khodabandelou, C. Hug and C. Salinesi, "A novel approach to process mining: Intentional process models discovery," in *2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS)*, IEEE, 2014, pp. 1-12.

[101] R. Deneckère, C. Hug, G. Khodabandelou and C. Salinesi, "Intentional process mining: discovering and modeling the goals behind processes using supervised learning," *International Journal of Information System Modeling and Design (IJISMD),* vol. 5, no. 4, pp. 22-47, 2014.

[102] G. Khodabandelou, C. Hug, R. Deneckère and C. Salinesi, "Unsupervised discovery of intentional process models from event logs," in *Proceedings of the 11th Working Conference on Mining Software Repositories*, ACM, 2014, pp. 282-291.

[103] G. Khodabandelou, C. Hug, R. Deneckère and C. Salinesi, "Supervised vs. Unsupervised Learning for Intentional Process Model Discovery," *Lecture Notes in Business Information Processing,* vol. 175 LNBIP, pp. 215-229, 2014.

[104] G. Khodabandelou, "Mining Intentional Process Models," in *Doctoral dissertation* , Paris, 2014.

[105] E. V. Epure, C. Hug, R. Deneckere and S. Brinkkemper, "What shall I do next? Intention mining for flexible process enactment.," in *International Conference on Advanced Information Systems Engineering*, Springer, 2014, pp. 473-487.

[106] G. Khodabandelou, C. Hug and C. Salinesi, "Mining users' intents from logs," *International Journal of Information System Modeling and Design (IJISMD),* vol. 6, no. 2, pp. 43-71, 2015.

[107] W. Krathu, R. Engel, C. Pichler, M. Zapletal and H. Werthner, "Identifying inter-organizational key performance indicators from EDIFACT messages," in *2013 IEEE 15th Conference on Business Informatics*, 2013.

[108] W. Krathu, C. Pichler, R. Engel, M. Zapletal, H. Werthner and C. Huemer, "A Framework for Inter-Organizational Performance Analysis from EDI Messages," in *2014 IEEE 16th Conference on Business Informatics*, 2014.

[109] M. Dees, M. de Leoni and F. Mannhardt, "Enhancing Process Models to Improve Business Performance: A Methodology and Case Studies," 2017, pp. 232-251.

[110] M. Cho, M. Song, M. Comuzzi and S. Yoo, "Evaluating the effect of best practices for business process redesign: An evidence-based approach based on process mining techniques," *Decision Support Systems,* vol. 104, pp. 92-103, dec 2017.

[111] H. Tran, U. Zdun and S. Dustdar, "Modeling human aspects of business processes–a view-based, model-driven approach," in *European Conference on Model Driven Architecture-Foundations and Applications*, 2008.

[112] J. Abraham and M. C. Reddy, "Challenges to inter-departmental coordination of patient transfers: a workflow perspective," *International Journal of medical informatics,* vol. 79, no. 2, pp. 112-122, 2010.

[113] R. H. Ballou, S. M. Gilbert and A. Mukherjee, "New managerial challenges from supply chain opportunities," *Industrial Marketing Management,* vol. 29, no. 1, pp. 7-18, 2000.

[114] J. Kingston, B. Schafer and W. Vandenberghe, "Towards a financial fraud ontology: A legal modelling approach," *Artificial Intelligence and Law,* vol. 12, no. 4, pp. 419-446, 2004.

[115] M. Georgeff and A. Rao, "Rational software agents: from theory to practice," in *Agent technology*, 1998, pp. 139-160.

[116] J. E. Cook and A. L. Wolf, "Automating process discovery through event-data analysis," in *ICSE '95 Proceedings of the 17th international conference on Software engineering*, ACM, 1995, pp. 73-82.

[117] M. Malinova, R. Dijkman and J. Mendling, "Automatic extraction of process categories from process model collections," in *International Conference on Business Process Management*, 2013.

[118] J. C. Buijs, B. van Dongen and W. van der Aalst, "Mining configurable process models from collections of event logs," in *11th International Conference, BPM 2013,*, 2013.

[119] A. Hallerbach, T. Bauer and M. Reichert, "Capturing variability in business process models: the Provop approach," *Journal of Software Maintenance and Evolution: Research and Practice,* vol. 22, no. 6-7, pp. 519-546, 2010.

[120] A. Lapouchnian, Y. Yu and J. Mylopoulos, "Requirements-driven design and configuration management of business processes," in *International Conference on Business Process Management*, 2007.

[121] G. Bernard and P. Andritsos, "A process mining based model for customer journey mapping.," in *Proceedings of the Forum and Doctoral Consortium Papers Presented at the 29th International Conference on Advanced Information Systems Engineering (CAiSE 2017)*, 2017.

[122] T. Mikolov, K. Chen, G. Corrado and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.

[123] E. C. Groen, N. Seyff, R. Ali, F. Dalpiaz, J. Doerr, E. Guzman, M. Hosseini, J. Marco, M. Oriol, A. Perini and others, "The crowd in requirements engineering: The landscape and challenges," *IEEE software,* vol. 34, no. 2, pp. 44-52, 2017.

[124] R. Snijders, F. Dalpiaz, M. Hosseini, A. Shahri and R. Ali, "Crowd-centric requirements engineering," in *IEEE/ACM 7th International Conference on Utility and Cloud Computing (UCC)*, IEEE, 2014, pp. 614-615.

[125] W. Maalej, M. Nayebi, T. Johann and G. Ruhe, "Toward data-driven requirements engineering," *IEEE Software*, vol. 33, no. 1, pp. 48-54, 2016.

[126] V. A. Rubin, A. A. Mitsyuk, I. A. Lomazova and W. M. P. van der Aalst, "Process mining can be applied to software too!," in *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM '14*, 2014.

[127] S. Ghanavati, D. Amyot and L. Peyton, "A systematic review of goal-oriented requirements management frameworks for business process compliance," in *2011 Fourth International Workshop on Requirements Engineering and Law*, 2011.

[128] J. C. Maxwell, A. I. Antón, P. Swire, M. Riaz and C. M. McCraw, "A legal cross-references taxonomy for reasoning about compliance requirements," *Requirements Engineering*, vol. 17, no. 2, pp. 99-115, jun 2012.

[129] N. Zeni, N. Kiyavitskaya, L. Mich, J. R. Cordy and J. Mylopoulos, "GaiusT: supporting the extraction of rights and obligations for regulatory compliance," *Requirements Engineering*, vol. 20, no. 1, pp. 1-22, mar 2015.

[130] E. Ramezani Taghiabadi, D. Fahland, B. F. van Dongen and W. M. P. van derAalst, "Diagnostic Information for Compliance Checking of Temporal Compliance Requirements," in *International Conference on Advanced Information Systems Engineering*, Springer, 2013, pp. 304-320.

[131] L. Thao, F. Maria, M. Montali, S. Rinderle-Ma and W. M. van der Aalst, "Compliance Monitoring in Business Processes: Functionalities, application, and tool-support. Information systems," *Information Systems*, vol. 54, pp. 209-234, 2015.

[132] M. El Kharbili, A. K. A. de Medeiros, S. Stein and W. van der Aalst, "Business Process Compliance Checking: Current State and Future Challenges," *MobIS*, pp. 107-113, 2008.

[133] S. Poelmans, "Coping strategies and distributed viscosity in a workflow management system: a case study," in *Workshop on Adaptive Workflow Systems.*, 1998.

[134] S. Poelmans, "Workarounds and distributed viscosity in a workflow system," *ACM SIGGROUP Bulletin*, vol. 20, no. 3, pp. 11-12, 1999.

[135] N. Outmazgin, "Exploring workaround situations in business processes," in *Lecture Notes in Business Information Processing*, 2013.

[136] D. M. Blei, B. B. Edu, A. Y. Ng, A. S. Edu, M. I. Jordan and J. B. Edu, "Latent Dirichlet Allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993-1022, 2003.

[137] X. Xu, T. Jin, Z. Wei, C. Lv and J. Wang, "TCPM: topic-based clinical pathway mining," in *2016 IEEE First International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, 2016, pp. 292-301.

[138] M. Dowson, "Iteration in the software process; review of the 3rd International Software Process Workshop," *Proceedings of the 9th international conference on Software Engineering*, pp. 36-41, 1987.

[139] E. V. Epure, C. Hug, R. Denckère and S. Brinkkemper, "Intention-mining: A solution to process participant support in process aware information systems," Department of Information and Computing Sciences, Utrecht University, 2013.

[140] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257-286, 1989.

[141] L. E. Baum, T. Petrie, G. Soules and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *The annals of mathematical statistics*, vol. 41, no. 1, pp. 164-171, 1970.

[142] L. Rabiner and B. Juang, "An introduction to hidden Markov models," *IEEE assp magazine*, vol. 3, no. 1, pp. 4-16, 1986.

[143] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-Means Clustering Algorithm," *Applied Statistics*, vol. 28, no. 1, p. 100, 1979.

[144] G. D. Forney, "The viterbi algorithm," *Proceedings of the IEEE,* vol. 61, no. 3, pp. 268-278, 1973.

[145] S. D. P. Flapper, L. Fortuin and P. P. Stoop, "Towards consistent performance management systems," *International Journal of Operations & Production Management,* vol. 16, no. 7, pp. 27-37, jul 1996.

[146] P. Hornix, "Performance analysis of business processes through process mining," 2007.

[147] D. Fahland and W. van der Aalst, "Model repair—aligning process models to reality," *Elsevier,* vol. 47, pp. 220-243, 2015.

[148] W. van der Aalst, A. Adriansyah and B. Van Dongen, "Replaying history on process models for conformance checking and performance analysis," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery,* vol. 2, no. 2, pp. 182-192, mar 2012.

[149] H. A. Reijers and S. L. Mansar, *Best practices in business process redesign: an overview and qualitative evaluation of sucessful redesign heuristics,* vol. 33, 2005, pp. 283-306.

[150] M. A. Hammer and J. Champy, "Reengineering the Corporation: A Manifesto for Business Revolution," *Business Horizons,* vol. 36, no. 5, pp. 90-91, 1993.

[151] Vienna University of Technology and the Eindhoven University of Technology, "EDImine - Mining Inter-organizational Business Processes," 2011. [Online]. Available: http://edimine.ec.tuwien.ac.at/. [Accessed 2018].

[152] M. A. Pitman, "Qualitative Research Design: An Interactive Approach," *Anthropology & Education Quarterly,* vol. 29, no. 4, pp. 499-501, 1998.

[153] R. Feldt and A. Magazinius, "Validity Threats in Empirical Software Engineering Research-An Initial Survey," in *Proceedings of the 22nd International Conference on Software Engineering and Knowledge Engineering*, 2010.

[154] M. Dumas, M. La Rosa, J. Mendling and H. A. Reijers, Fundamentals of business process management, Springer, 2013.

[155] M. Vidgof, D. Djurica, S. Bala and J. Mendling, "Cherry-Picking from Spaghetti: Multi-range Filtering of Event Logs," in *Enterprise, Business-Process and Information Systems Modeling*, 2020.

[156] J. De Weerdt, S. Vanden Broucke, J. Vanthienen and B. Baesens, "ctive trace clustering for improved process discovery," *IEEE Transactions on Knowledge and Data Engineering,* vol. 25, no. 12, pp. 2708--2720, 2013.

[157] A. Augusto, R. Conforti, M. Dumas, M. La Rosa, F. M. Maggi, A. Marrella, M. Mecella and A. Soo, "Automated discovery of process models from event logs: Review and benchmark," *IEEE transactions on knowledge and data engineering,* vol. 31, no. 4, pp. 686--705, 2018.

[158] R. Conforti, M. La Rosa and A. H. ter Hofstede, "Filtering Out Infrequent Behavior from business process event logs," *IEEE Transactions on Knowledge and Data Engineering,* vol. 29, no. 2, pp. 300-314, 2016.

[159] M. Fani Sani, S. van Zelst and W. van der Aalst, "Improving process discovery results by filtering outliers using conditional behavioural probabilities," in *International Conference on Business Process Management*, 2017.

[160] R. A. G. Oliva, J. . J. S. Barrenechea and A. A. A. Jimmy , "Control Metrics Evaluation Model for Business Processes using Process Mining," in *The Tenth International Conference on Information, Process, and Knowledge Management*, 2018.

[161] R. Conforti, M. La Rosa and A. Ter Hofstede, "Timestamp Repair for business process event logs," 2018.

[162] J. Brzozowski, G. Jirásková and C. Zou, "Quotient Complexity of Closed Languages," *Theory of Computing Systems,* vol. 54, no. 2, pp. 277--292, 2014.

[163] "ProM Tools," [Online]. Available: https://www.promtools.org/doku.php?id=docs:objects:event_filter#simple_event_filters. [Accessed 04 2021].

[164] A. K. A. De Medeiros, A. Guzzo, G. Greco, W. M. Van der Aalst, A. Weijters, B. F. Van Dongen and D. Sacca, "Process mining based on clustering: A quest for precision," in *International conference on business process management*, 2007.

[165] G. Greco, A. Guzzo, L. Pontieri and D. Sacca, "Discovering Expressive Process Models by clustering log traces," *IEEE Transactions on knowledge and data engineering*, vol. 18, no. 8, pp. 1010--1027, 2006.

[166] R. J. C. Bose and W. M. Van der Aalst, "Context aware trace clustering: Towards improving process mining results," in *International Conference on Data Mining*, 2009.

[167] P.-N. Tan, M. Steinbach and V. Kumar, Introduction to data mining, Pearson Education India, 2016.

[168] R. J. C. Bose and W. M. van der Aalst, "Trace clustering based on conserved patterns: Towards achieving better process models," in *International Conference on Business Process Management*, 2009.

[169] G. M. Veiga and D. R. Ferreira, "Understanding spaghetti models with sequence clustering for ProM," in *International conference on business process management*, 2009.

[170] J. R. Venable, J. Pries-Heje and R. L. Baskerville, "Choosing a Design Science Research Methodology," in *Australasian Conference on Information Systems*, 2017.

[171] K. Pfeffers, T. Tuunanen, C. E. Gengler, M. Rossi, W. Hui, V. Virtanen and J. Bragge, "The design science research process: A model for producing and presenting information systems research," in *Proceedings of the First International Conference on Design Science Research in Information Systems and Technology*, 2006.

[172] A. R. Hevner, S. T. March, J. Park and S. Ram, "Design Science in Information Systems Research," *MIS Quarterly*, vol. 28, no. 1, pp. 75-105, 2004.

[173] S. Gregor and A. Hevner, "Positioning and Presenting Design Science Research for Maximum Impact," *MIS Quarterly*, vol. 37, no. 2, pp. 337--355, 2013.

[174] W. van der Aalst, Process Mining Data Science in Action, 2 ed., Springer, 2016.

[175] D. Giannakopoulou and K. Havelund, "Automata-based verification of temporal properties on running programs," in *Proceeding of 16th International Conference on Automated Software Engineering*, 2001.

[176] W. van der Aalst, H. De Beer and B. F. van Dongen, "Process Mining and Verification of Properties: An Approach Based on Temporal Logic," in *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, 2005, pp. 130-147.

[177] J. Horkoff, D. Barone, L. Jiang, E. Yu, D. Amyot, A. Borgida and J. Mylopoulos, "Strategic business modeling: representation and reasoning," *Software & Systems Modeling*, vol. 13, no. 3, pp. 1015-1041, 2014.

[178] "jUCMNav," 2016. [Online]. Available: http://softwareengineering.ca/jucmnav.

[179] Y. Fan, A. A. Anda and D. Amyot, "An Arithmetic Semantics for GRL Goal Models with Function Generation," in *10th International Conference on System Analysis and Modeling*, 2018.

[180] L. Wen, W. M. van der Aalst, J. Wang and J. Sun, "Mining process models with non-free-choice constructs," *Data Mining and Knowledge Discovery*, vol. 15, no. 2, pp. 145-180, 2007.

[181] IBM, "IBM CPLEX Optimizer," [Online]. Available: https://www.ibm.com/analytics/cplex-optimizer. [Accessed 2021].

[182] N. L. Johnson, S. Kotz and N. Balakrishnan, "Beta distributions," *Continuous univariate distributions*, vol. 2, pp. 221-235, 1994.

[183] F. Mannhardt and D. Blinde, "Analyzing the Trajectories of Patients with Sepsis using Process Mining," in *RADAR+EMISA@ International Conference on Advanced Information Systems Engineering, CAiSE* , 2017.

[184] "Sepsis Cases - Event Log," Eindhoven University of Technology, [Online]. Available: https://data.4tu.nl/articles/dataset/Sepsis_Cases_-_Event_Log/12707639. [Accessed 6 2021].

[185] S. J. Leemans, D. Fahland and W. M. van der Aalst, "Discovering block-structured process models from event logs-a constructive approach," in *International conference on applications and theory of Petri nets and concurrency*, 2013.

[186] S. J. Leemans, D. Fahland and W. M. van der Aalst, "Discovering Block-Structured Process Models from Event Logs Containing Infrequent Behaviour," in *International conference on business process management*, 2013.

[187] G. Sedrakyan, J. De Weerdt and M. Snoeck, "Process-mining enabled feedback:“tell me what I did wrong” vs.“tell me how to do it right”," *Computers in human behavior*, vol. 57, pp. 352--376, 2016.

[188] T. Gurgen Erdogan and A. Tarhan, "A goal-driven evaluation method based on process mining for healthcare processes," *Applied Sciences*, vol. 8, no. 6, p. 894, 2018.

[189] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Machine learning*, vol. 23, no. 1, pp. 69-101, 1996.

[190] P. Runeson, M. Host, A. Rainer and B. Regnell, Case study research in software engineering: Guidelines and examples, Wiley, 2012.

[191] P. Spoletini and A. Ferrari, "Requirements Elicitation: A Look at the Future through the Lenses of the Past," in *25th IEEE International Requirements Engineering Conference*, IEEE CS, 2017, pp. 476-477.

[192] E. C. Groen, J. Doerr and S. Adam, "Towards crowd-based requirements engineering a research preview," in *International Working Conference on Requirements Engineering: Foundation for Software Quality*, Springer, 2015, pp. 247-253.

[193] A. Sutcliffe, "Scenario-based requirements engineering," in *11th IEEE International Requirements Engineering Conference. (RE03)*, IEEE CS, 2003, pp. 320-329.

[194] H. Saiedian, P. Kumarakulasingam and M. Anan, "Scenario-based requirements analysis techniques for real-time software systems: a comparative evaluation," *Requirements Engineering*, vol. 10, no. 1, pp. 22-33, 2005.

[195] A. G. Sutcliffe, N. A. Maiden, S. Minocha and D. Manuel, "Supporting scenario-based requirements engineering," *IEEE Transactions on software engineering*, vol. 24, no. 12, pp. 1072-1088, 1998.

[196] W. van der Aalst, K. van Hee and J. v. W. Computer, "Auditing 2.0: Using process mining to support tomorrow's auditor," *Computer*, vol. 43, no. 3, 2010.

Appendix A: Event Log of the Illustrative Example

Case ID	Task label	Task Name	Event Type	org:re-source	Resource Label	Timestamp
1	a	admission	completed	Sarah	Nurse	12/03/2017 9:35
2	a	admission	completed	Sarah	Nurse	12/03/2017 9:45
1	b	regular test	completed	Tina	Lab Assistant	12/03/2017 11:12
2	b	regular test	completed	Tina	Lab Assistant	12/03/2017 11:32
3	a	admission	completed	Rose	Nurse	13/03/2017 9:02
4	a	admission	completed	Sarah	Nurse	13/03/2017 9:10
3	b	regular test	completed	Tina	Lab Assistant	13/03/2017 9:14
5	a	admission	completed	Rose	Nurse	13/03/2017 10:11
5	b	regular test	completed	Tina	Lab Assistant	13/03/2017 11:12
4	b	regular test	completed	Tina	Lab Assistant	13/03/2017 11:45
2	c	check the result	completed	Hannah	Physician	15/03/2017 11:39
3	c	check the result	completed	Hannah	Physician	15/03/2017 11:40
2	d	request for advanced	completed	Hannah	Physician	15/03/2017 11:41
1	c	check the result	completed	Hannah	Physician	15/03/2017 13:01
4	c	check the result	completed	Hannah	Physician	15/03/2017 14:25
4	f	request for repetition	completed	Hannah	Physician	15/03/2017 14:36
3	f	request for repetition	completed	Hannah	Physician	15/03/2017 14:40
1	g	send the result	completed	Jane	Nurse	16/03/2017 13:01
3	b	regular test	completed	Tina	Lab Assistant	17/03/2017 9:08
4	b	regular test	completed	Tina	Lab Assistant	18/03/2017 9:01
5	c	check the result	completed	Hannah	Physician	18/03/2017 12:35
5	d	request for advanced	completed	Hannah	Physician	18/03/2017 12:48
3	c	check the result	completed	Hannah	Physician	18/03/2017 14:25
3	g	send the result	completed	Jane	Nurse	19/03/2017 12:11
2	e	advanced test	completed	Linda	Lab Assistant	20/03/2017 9:35
2	c	check the result	completed	Hannah	Physician	22/03/2017 14:12
4	c	check the result	completed	Hannah	Physician	22/03/2017 14:35
4	d	request for advanced	completed	Hannah	Physician	22/03/2017 14:51
5	b	regular test	completed	Tina	Lab Assistant	23/03/2017 10:11
2	g	send the result	completed	Jane	Nurse	23/03/2017 14:15
4	e	advanced test	completed	Tina	Lab Assistant	25/03/2017 11:11
5	c	check the result	completed	Hannah	Physician	29/03/2017 11:11
5	d	request for advanced	completed	Hannah	Physician	29/03/2017 11:21
4	c	check the result	completed	Hannah	Physician	29/03/2017 15:01
4	g	send the result	completed	Jane	Nurse	30/03/2017 14:15
5	e	advanced test	completed	Linda	Lab Assistant	03/04/2017 11:12
5	c	check the result	completed	Hannah	Physician	08/04/2017 12:35
5	g	send the result	completed	Jane	Nurse	13/04/2017 15:12