KU LEUVEN
FACULTEIT ECONOMIE EN
BEDRIJFSWETENSCHAPPEN

Business Process Discovery: New Techniques and Applications

Proefschrift voorgedragen tot
het behalen van de graad van
Doctor in de Toegepaste
Economische Wetenschappen

door

Jochen De Weerdt

Nummer 403 2012

Committee

prof. dr. Marleen Willekens (*Chair*)	KU Leuven
prof. dr. Bart Baesens (*Promotor*)	KU Leuven
prof. dr. Jan Vanthienen (*Co-Promotor*)	KU Leuven
prof. dr. Manu De Backer	Universiteit Antwerpen
prof. dr. ir. David Martens	Universiteit Antwerpen
prof. dr. Jan Mendling	Wirtschaftsuniversität Wien
prof. dr. ir. Wil M.P. van der Aalst	Technische Universiteit Eindhoven

To Charlot.

Table of Contents

Samenvatting

Tegenwoordig slaan bedrijven en andere organisaties gigantische volumes aan elektronische gegevens op. Dit komt omdat dataopslag steeds goedkoper wordt en omdat men ex-post wil of moet kunnen nagaan waar fouten, problemen, enz. zijn voorgekomen in de bedrijfsvoering. Echter, er is hoofdzakelijk geïnvesteerd in dataverzameling en -opslag terwijl investeringen in het effectief analyseren van de data achterop hinkt. Daarbij komt dat bedrijfsprocessen in het algemeen meer en meer geautomatiseerd worden wat ervoor zorgt dat het steeds makkelijker wordt om data over de uitvoering ervan bij te houden. Deze automatisering wordt gestuurd door informatiesystemen die de bedrijfsprocessen coördineren en faciliteren. Wanneer deze informatiesystemen op een gestructureerde manier bijhouden welke stappen in een bedrijfsproces worden uitgevoerd ontstaat er een interessante opportuniteit om inzicht te verwerven in hoe bedrijfsprocessen *in werkelijkheid* worden uitgevoerd. Dat dergelijke analyse meer en meer aan relevantie wint wordt versterkt door het feit dat automatisering ook met zich meebrengt dat het moeilijker is om een goed overzicht te behouden over het end-to-end proces.

De nood aan concrete inzichten om de operationele bedrijfsvoering te optimaliseren heeft tot gevolg dat data-analysetechnieken aan relevantie winnen. *Process Mining* is een relatief jong onderzoeksdomein dat zich kenmerkt door een specifieke focus op de ontwikkeling en het gebruik van algoritmes en technieken voor de analyse van executiedata van allerhande bedrijfsprocessen. De doelstelling van Process Mining is om niet-triviale kennis te ontdekken op basis van de data die opgeslagen worden door informatiesystemen met als doel de bedrijfsprocessen te verbeteren. Eén van de belangrijke taken in het domein is *process discovery*, met name de automatische constructie van een procesmodel dat weergeeft hoe een bedrijfsproces er in werkelijkheid uitziet en dit op basis van de data in een event log.

Process Mining: Op het raakvlak tussen BPM and KDD

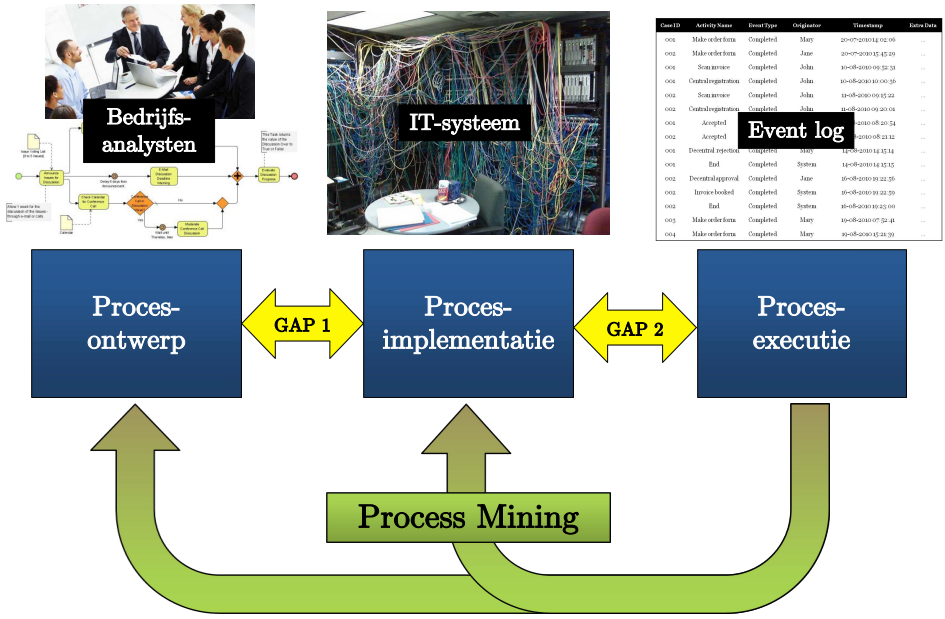
Process Mining bevindt zich op het raakvlak tussen twee ruimere onderzoeksdomeinen, meer bepaald Business Process Management (BPM) en Knowledge Discovery in Databases (KDD). Business Process Management (BPM) is de verzamelnaam voor technieken, concepten en tools voor het ontwerp, de uitvoering, de controle en de analyse van bedrijfsprocessen binnen een organisatie. Process Mining-technieken kunnen een cruciale rol spelen in de vervolledigingvan de levenscyclus van een typisch bedrijfsproces, met name wat betreft de analyse en verbetering van een bepaalde procesimplementatie. Merk op dat Process Mining in die zin veel verder gaat dan traditionele Business Intelligence (BI). Een Process Mining-analyse start bottom-up, vertrekkende van de executiedata, en is veel ingrijpendere dan bijvoorbeeld de evaluatie van KPI's. Behalve met het BPM-domein heeft Process Mining ook veel gemeenschappelijk met het KDD-veld. Knowledge Discovery in Databases omvat het niet-triviale proces om geldige, nieuwe, potentieel bruikbare en verstaanbare patronen te ontdekken in data. Hierbij spelen data mining-technieken een cruciale rol omdat ze in staat zijn om net dit soort patronen te vinden. Zowel onderzoek naar als het gebruik van dergelijke technieken is in het laatste decennium sterk toegenomen omdat bedrijven met sterke data-analysecapaciteiten een significant competitief voordeel kunnen creëren.

Process Mining blijkt in die mate nuttig omdat de technieken twee belangrijke hiaten in de levenscyclus van een bedrijfsproces kunnen bloot leggen. Meer bepaald maakt Process Mining het mogelijk om de verschillen tussen hoe businessanalisten het proces uittekenen en hoe de werkelijke implementatie gebeurt te expliciteren. Ten tweede stelt het analisten ook in staat om de uitvoering door eindgebruikers te contrasteren met de systeemconfiguratie en het procesontwerp. De kennis die op die manier beschikbaar wordt, blijkt van onschatbare waarde voor procesexperts en managers. Figuur 1 illustreert de toegevoegde waarde die Process Mining kan bieden door patronen te visualiseren en specifieke kennis over de werkelijke procesexecutie naar boven te brengen zodat deze informatie kan gebruikt worden om zowel het ontwerp als de implementatie van het bedrijfsproces te verbeteren.

De bijdragen die in deze thesis worden voorgesteld, kunnen volledig gesitueerd worden binnen het domein van Process Mining. De thesis bevat vier belangrijke onderdelen waarbij, *process discovery*, zijnde het automatisch ontdekken van procesmodellen, het centrale thema vormt.

Metrieken voor het evalueren van ontdekte procesmodellen

Het eerste deel van dit proefschrift behandelt het probleem van de evaluatie van de kwaliteit van ontdekte procesmodellen. De kwaliteit van een ontdekt proces-



Figuur 1: Het nut van Process Mining geïllustreerd

model kan worden beoordeeld vanuit verschillende perspectieven. Op het hoogste niveau valt de evaluatie van een procesmodel uiteen in twee elementen: accuraatheid en begrijpbaarheid. Onze bijdrage is tweeledig. Allereerst wordt een analyse uitgevoerd van de kwaliteit van bestaande evaluatiemetrieken. Ten tweede wordt een nieuwe precisiemetriek voorgesteld, gebaseerd op de idee om artificiële negatieve events te induceren in een een event log. Deze metriek ligt aan de basis voor de toepassing van de F-score voor de evaluatie van een ontdekt procesmodel. We beschouwen deze nieuwe evaluatiemethode als een eerste stap in de richting van een omvattend evaluatiekader voor procesontdekkingstechnieken.

Onze bijdragen werden beschreven en gepubliceerd in:

- J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens. **A Critical Evaluation Study of Model-Log Metrics in Process Discovery**. In M. zur Muehlen and J. Su, editors, *Business Process Management Workshops*, 66 of *Lecture Notes in Business Information Processing*, pages 158–169. Springer, 2010.

- J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens. **A robust F-measure for evaluating discovered process models**. In *CIDM*, pages 148–155. IEEE, 2011.

Evalueren van procesontdekkingstechnieken aan de hand van real-life procesgegevens

In het domein van Process Mining wordt er traditioneel gebruik gemaakt van artificiële gegevens bij de ontwikkeling en evaluatie van procesontdekkingstechnieken. Dit is logisch omwille van het feit dat onderzoekers op die manier eenvoudig de kwaliteit van de technieken kunnen evalueren en omdat men speciale proceseigenschappen kan simuleren waarmee procesontdekkingstechnieken het moeilijk hebben. Het is echter zo dat het gebruik van levensechte datasets voor de evaluatie van dergelijke technieken vrij beperkt is. In deze thesis wordt daarom specifiek de nadruk gelegd op de evaluatie van procesontdekkingstechnieken vanuit verschillende perspectieven op basis van acht real-life event logs. Hierbij wordt gefocust op accuraatheid, begrijpbaarheid en schaalbaarheid van de algoritmes.

Dit onderdeel van de thesis werd gepubliceerd in:

- J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens. **A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs**. *Information Systems*, 37(7):654–676, 2012.

Actief clusteren van procesinstanties voor betere procesontdekkingsresultaten

Eén van de belangrijkste en tegelijkertijd moeilijkste uitdagingen in het Process Mining-domein bestaat erin om accurate en begrijpbare modellen te ontdekken uit procesgegevens die opgeslagen worden tijdens de uitvoering van bedrijfsprocessen in zeer flexibele omgevingen. Door het beperkte aantal restricties dat wordt opgelegd aan de actoren in het bedrijfsproces, merken we dat deze flexibiliteit leidt tot een zeer grote variëteit aan manieren waarop procesinstanties worden uitgevoerd. Dit zorgt er meteen ook voor dat de toepassing van procesanalyse net de grootste meerwaarde creëert. Echter, het eenvoudigweg toepassen van procesontdekkingstechnieken op dit soort gegevens leidt vaak tot onnauwkeurige en onbegrijpbare procesmodellen. Het clusteren van procesinstanties in meerder subgroepen is een interessante piste om dit probleem aan te pakken. Door te clusteren wordt het mogelijk om één volledige event log op te splitsen in meerdere datasets zodat het makkelijker wordt voor ontdekkingstechnieken om accurate en begrijpbare modellen te construeren. In deze thesis beschrijven we een nieuw algoritme, *ActiTraC* genaamd, dat sterk verschilt van eerder voorgestelde algoritmes in de literatuur. Onze techniek vertrekt van de observatie dat bestaande technieken te lijden hebben onder het feit dat er een groot verschil is tussen de manier waarop ze clusteren en de wijze van evaluatie van de clusteroplossing. *ActiTraC* overbrugt de kloof tussen clusteren en evalueren door het toepassen van een “active

learning"-geïnspireerde aanpak waarbij procesinstanties iteratief worden toegevoegd aan clusters op basis van het feit of de cluster voldoende accuraat blijft na toevoeging. Uit uitgebreide experimentele analyses blijkt dat *ActiTraC* in staat is om zowel op het vlak van accuraatheid als op het vlak van begrijpbaarheid beter te scoren dan bestaande clusteringtechnieken.

Dit onderdeel van de thesis is in submissie:

- J. DE WEERDT, S. VANDEN BROUCKE, J. VANTHIENEN, AND B. BAESENS. **Active Trace Clustering for Improved Process Discovery**. *Submitted to IEEE Transactions on Knowledge and Data Engineering*, 2012.

Bedrijfskundige toepassingen

Tot slot bevat deze thesis ook een beschrijving van verschillende gevalstudies die de kracht van bedrijfsprocesanalyses aan de hand van Process Mining-technieken demonstreert.

Deze studies zij beschreven en gepubliceerd in:

- J. DE WEERDT, A. SCHUPP, A. VANDERLOOCK, AND B. BAESENS. **Process Mining for the multi-faceted analysis of business processes - A case study in a financial services organization**. *Computers in Industry*, Forthcoming.

- S. GOEDERTIER, J. DE WEERDT, D. MARTENS, J. VANTHIENEN, AND B. BAESENS. **Process discovery in event logs: An application in the telecom industry**. *Appl. Soft Comput.*, 11(2):1697–1710, 2011.

- J. DE WEERDT, F. CARON, J. VANTHIENEN, AND B. BAESENS. **Getting a Grasp on Clinical Pathway Data: An Approach Based on Process Mining**. In *The Third Workshop on Data Mining for Healthcare Management at the 16th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Kuala Lumpur, Malaysia (forthcoming)*, 2012.

- J. DE WEERDT, S. VANDEN BROUCKE, J. VANTHIENEN, AND B. BAESENS. **Leveraging process discovery with trace clustering and text mining for intelligent analysis of incident management processes**. In *IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2012.

Chapter 1

Introduction

Within today's organizations, data are being collected and accumulated at a drastic pace. Although a lot of investments have been made in data collection and storage, similar investments in analyzing these data remain behind. Especially with respect to business processes, automation is increasing incessantly which causes a significant growth in the availability of process-related data. Within and across departments, information systems have been put in place in order to coordinate and facilitate the business processes. Since these systems keep track of every-day business transactions, important opportunities arise for analysis of process-related data so as to provide insight into the actual way of working. Moreover, automation often causes an increased complexity with a reduced overview of the overall end-to-end process. Because of the lack of sensible overview and tangible insights, there exist vast opportunities for data analytics, a domain which can be best defined as a broad set of intelligent techniques for gaining insights from data. When the focal point of these intelligent techniques is business processes, the field of research is often termed Process Mining. To be precise, the key objective of Process Mining is the extraction of non-trivial knowledge from event data as recorded by information systems.

This introductory chapter starts with a detailed outline of the field of Process Mining. Furthermore, the role of intelligent process analysis techniques within the Business Process Management (BPM) domain is discussed. In addition, we position the main learning tasks of the process mining domain with respect to the principles of the broader field of Knowledge Discovery in Databases (KDD). Finally, the contributions of this thesis are set forth.

1.1 The Field of Process Mining

Process Mining is a relatively young area of academic research. The main goal of process mining techniques is to employ process-related data in order to extract information and knowledge, for instance by automatically discovering a process model. In this section, the concept of an event log, which is the cornerstone of analysis, is detailed. Furthermore, the main building blocks of the process mining domain are discussed.

1.1.1 The Event Log as the Cornerstone of Analysis

As in many other knowledge discovery domains, data are the crucial building block. Within Process Mining, data are accounted for by the concept of an event log. A sample event log is shown in Figure 1.1. In the context of such event logs, an event is defined as the registration of an activity instance state change. Activity instances are the actual units of work as they represent the actual work conducted in the context of a certain process instance. States of activity instances are specified by the enactment language of a certain process modeling paradigm. Different process modeling paradigms present distinct execution semantics which are defined in terms of state transition diagrams. A very simple state transition diagram is shown in Figure 1.2. In the process modeling literature, much more elaborate state transition diagrams are described. For instance, the Object Management Group (OMG) defines the *life cycle of an activity* in their Business Process Modeling Notation (BPMN) specification [100]. Furthermore, other process modeling paradigms such as YAWL (Yet Another Workflow Language), case handling [144] and declarative approaches (EM-BrA^{2}CE [61], Declare [102]) propose distinct but similar diagrams for their respective enactment semantics.

Although it is interesting to sift through these different enactment semantics and state transition diagrams, for process mining analysis, the actual data are crucial. In practice, we often see that process data are pooled from different repositories originating from CRM, ERP, WfM and other types of information systems. As a result, a clear correspondence between the theoretical foundation of the concept of an event in terms of a state transition diagram and the data found in practice, is difficult to distinguish. In many flexible environments such as financial services, customer relationship management, product development, etc. business processes rely on legacy systems or less process-oriented information systems. Frequently, the registration of the business events is coarse-grained such that only one type of state transition (e.g. completion of an activity instance) can be extracted from the actual data. Since Process Mining can be considered most useful in these flexible environment where systems typically show a much larger variety of behavior, the translation of the available data sources into an event log

Case ID	Activity Name	Event Type	Originator	Timestamp	Extra Data
001	Make order form	Start	Employee A	20-07-2012 14:02:06	...
001	Make order form	Complete	Employee A	20-07-2012 14:28:29	...
001	Scan invoice	Complete	Employee E	10-08-2012 09:52:31	...
001	Central registration	Start	Employee B	10-08-2012 10:00:36	...
001	Central registration	Complete	Employee B	11-08-2012 10:15:22	...
002	Make order form	Start	Employee C	13-08-2012 09:20:01	...
001	Accept invoice	Complete	Employee A	13-08-2012 09:20:54	...
002	Make order form	Complete	Employee D	13-08-2012 09:21:12	...
001	Reject invoice	Complete	Employee A	14-08-2012 14:15:14	...
001	End	Complete	System	14-08-2012 14:15:15	...
003	Make order form	Start	Employee A	16-08-2012 19:22:56	...
003	Make order form	Complete	Employee A	16-08-2012 19:22:59	...
002	Scan Invoice	Complete	Employee E	16-08-2012 19:23:00	...
003	Central registration	Start	Employee D	19-08-2012 07:52:41	...
003	Scan invoice	Complete	Employee E	19-08-2012 15:21:39	...
002	Central registration	Start	Employee E	19-08-2012 15:21:40	...
003	Central registration	Complete	Employee D	21-08-2012 09:09:39	...
002	Central registration	Complete	Employee D	21-08-2012 09:15:35	...
002	Request for change	Start	Employee A	21-08-2012 15:01:39	...
003	Accept invoice	Complete	Employee A	22-08-2012 14:21:00	...
003	...	...	...	...	...

Figure 1.1: An example of a typical event log

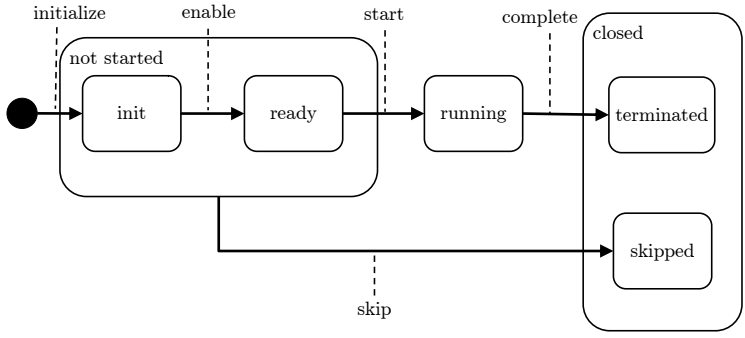


Figure 1.2: A simple state transition diagram

is often a non-trivial task. Amongst others, this is due to the lagging adoption of Business Process Management Systems (BPMSs) in practice.

The actual collection of event logs from information system's repositories is often carried out by means of text-based data formats. In reality, process data are scattered over different data sources and it can be a tedious task to define the

exact scope of the process under study. As a consequence, an extensive ETL-phase is required more often than not before any concrete analysis can be initiated. Also, the extracted data should be converted into a typical event log storage format. The earliest approach for storing event logs is MXML (Mining eXtensible Markup Language). Since 2003, this format used to be the de facto standard since it was tightly coupled with the ProM-framework, an academic tool for Process Mining. Only recently, a successor of MXML was adopted by the IEEE Task Force on Process Mining. This new format, called XES (eXtensible Event Stream), improves the original standard because it is less restrictive. The meta-model of XES is described in [69].

As event logs are the cornerstone for Process Mining, it is important to state the different requirements to which an event log should comply. The following three assumptions are crucial:

- events must refer to a well-defined step (activity instance) in a process instance identified by a unique activity name
- events must refer to distinct process instances or cases by making use of an ID
- events must be recorded in a totally ordered way, preferably by containing a timestamp

A formal definition of an event log. As stated, an event log consists of events that pertain to process instances. A process instance is defined as a logical grouping of activities whose state changes are recorded as events. We thus assume that it is possible to record events such that each event refers to a task (a step in a process instance), a case (the process instance) and that events are totally ordered. As such, let X be a set of event identifiers, P a set of case identifiers, the alphabet A a set of activity types and the alphabet E a set of event types corresponding with activity life cycle state transitions (e.g. *start*, *assign*, *skip*, *complete*, etc.). An event can then be formulated as a predicate $Event(x, p, a, e, t)$ with $x \in X$ the event identifier, $p \in P$ the case identifier and $a \in A$ the activity type, $e \in E$ the event type and $t \in \mathbb{N}$ the position of the event in its process sequence. Note that events commonly store much more information (timestamps, originators, case data, etc.), but for the sake of simplicity of the formal definition, this additional information is left out. Furthermore, it will be assumed that the state transition e for each logged event equals *complete*. The function $Case \in X \cup L \mapsto P$ denotes the case identifier of an event or sequence. The function $Activity \in X \mapsto A$ denotes the activity type of an event.

Let event log L be a set of process instances. Let $pi \in L$ be a process instance or event sequence; $pi = \{x | x \in X \wedge Case(x) = Case(pi)\}$. The function $Position \in$

$X \times L \mapsto \mathbb{N}_0$ denotes the position of an event in the corresponding process instance. The set X of event identifiers has a complete ordering, such that $\forall x, y \in X : x < y \vee y < x$ and $\forall x, y \in pi, x < y : Position(x) < Position(y)$. Let $x.y \subseteq pi$ be two subsequent event identifiers within a process instance pi; $x.y \Leftarrow \exists x, y \in pi : x < y \wedge \nexists z \in pi : x < z < y$. We will use this predicate in the context of single sequences which is therefore left implicit. Furthermore, a sequence of two events $x.y$ with activity types a and b respectively can be abbreviated as $\langle a, b \rangle$. Each pi_i in the event log now represents a different execution sequence, corresponding to a particular process instance, and can be depicted as $\langle a, b, c, ..., z \rangle$ with $a, b, c, ..., z$ the activity types of the ordered events contained in the sequence.

Finally, note that identical process instances (i.e. traces with the same execution sequence) can be grouped into a distinct process instance, further denoted as *dpi*. A *dpi* is defined as a set of *pi* with the number of *pi* in the set denoted as the frequency of the *dpi*. A collection of *dpi*'s is called a grouped event log (GL).

1.1.2 Building Blocks of Process Mining

As soon as the intensive phase of data extraction and transformation has resulted in a satisfactory event log, diagnosis based on Process Mining can be embarked on. Based on the ideas in [129] and [130], the fundamental building blocks of the process mining field are outlined, as represented in Table 1.1. In order to properly characterize the different process mining activities, two orthogonal dimensions can be distinguished: the log information dimension and the process mining task dimension. The former dimension refers to four different types of information perspectives from which an event log can be looked at. Firstly, the data can be assessed according to a control-flow view. Here the main focus is on the sequential relations between the different activities in the event log. Furthermore, organizational aspects can be at the center of attention. Accordingly, the analysis focuses on who is performing the different activities in the process and how these so-called originators are related. Thirdly, the case data view is concerned with extra data elements that might be available in the event log. Song et al. [121] bring up that this dimension is the major focus of existing business intelligence and data mining tools. Finally, data about exceptional behavior can be investigated as well, for example to find out how often or why certain activity instances were aborted.

Orthogonal to the log information dimension, the process mining task dimension can be identified. The partitioning of process mining activities and techniques according to more high-level process mining tasks involves the categorization into three different groups: discovery, conformance and/or compliance and enhancement tasks.

Discovery tasks search for underlying models and knowledge, answering questions such as “How are processes actually executed?”, “Which types of roles exist

Process mining task \ Log information	**Control-flow**	**Resource flow**	**Case data flow**	**Exception handling**
Discovery	process discovery	organizational model, social network	mining data integrity constraints	exception handling pattern detection
Conformance/ Compliance	delta analysis, conformance checking	four-eyes principle, Chinese wall	fraud detection	rework validation
Enhancement	performance bottleneck detection, time prediction	resource utilization analysis, organizational optimization	decision mining	exception impact analysis

Table 1.1: Business process mining field

in the process?", "How are people working together?" and "Who can change certain case data elements?". In the literature, especially the control-flow aspect has received a lot of attention. This subdomain termed *process discovery* accounts for a majority of published research papers. With its objective to visualize a collection of process executions in a process model, a wide range of different algorithms have been conceived, from the original α-algorithm to evolutionary computation-inspired approaches. In recent years however, also the resource dimension has become the subject of intelligent discovery techniques. For instance, the discovery of organizational structures and social networks between the involved performers can provide interesting insights into the actual business process. Note that the discovery of access control rules fits within this task dimension as well. Finally, discovery tasks can also be identified within the case data flow or exception handling dimensions. Very useful information about cases can be found in their according data elements, although this information is mainly useful for compliance and/or enhancement tasks.

Besides discovery tasks, one of the major practical benefits of process mining techniques lies within conformance and compliance checking tasks. With respect to this type of tasks, it is important to make a distinction between a narrow interpretation of conformance checking and the broader domain of compliance verification. These tasks are definitely intertwined because they both entail the use of comparative analysis procedures. However, conformance testing is defined as a set of quantification measures to assess the fit between an event log and a (process) model. As such, measuring the quality of a discovered process model based on the event log can also be categorized as conformance checking [96, 107], although it is more correct to term this task as process discovery evaluation rather

than conformance checking.

Comparative analyses can be looked at in a broader context as well. For instance, verifying whether actual process behavior as recorded in event logs complies with management expectations, formal business rules and/or legislative regulations is considered extremely valuable. These tasks can be categorized as compliance verification. Both formal compliance verification (regulatory compliance) as well as informal compliance verification (management expectations) are enabled through the use of process mining techniques. To this, it can be added that process mining techniques can be applied for auditing purposes as well. For instance, traditional principles such as the Chinese wall, the four-eyes principle, and segregation of duties can be verified. Also, Process Mining is an ideal means for fraud detection within a process-aware information system since it uses the actual information registered in the system's logs. Both the detection of formal and informal inconsistencies between observed and desired behavior can be an important starting point for process improvement. The severity of the discrepancies can be measured and explanations to support business process improvement initiatives can be outlined.

Finally, the diagnostic information that can be derived from an event log allows for a third type of tasks: enhancement tasks. Here, the goal is to enrich a process, organizational or social model with other information. Bottleneck analysis for example scrutinizes timestamp information of activities, process paths, performers and organizational units. Another example study fitting within this task dimension is a study by van der Aalst et al. [140] that elaborates on the prediction of the future of a running instance. The authors show that it is possible to answer questions like "When will this instance be completed?" or "What is the probability that this instance will undergo activity A?" in a very efficient way. Process Mining can thus be used in a real-time setting as well. Furthermore, Rozinat and van der Aalst [113] outline decision mining which approaches a control-flow process model as a set of decision points where the analysis tries to reveal the correlation between the execution path and certain data attributes of the process instance at hand.

1.2 Business Process Management

Business Process Management (BPM) is the collective term to designate concepts, methods and techniques to support the design, configuration, enactment, and analysis of business processes. The foundation of BPM is the explicit representation of a business process in some kind of model in order to define the activities and execution constraints between them. The cradle of BPM is process modeling, which refers to the identification and specification of business processes.

The early foundations of Business Process Management can be attributed to Hammer and Champy [70]. With their introduction of a radical approach to business process reengineering, they brought process orientation to the agenda. Furthermore, the role of information technology in reengineering business processes was pointed out by Davenport [36]. Information systems, supported by the plethora of information and communication technologies, support the core business processes in most of today's organizations. As such, any approach to improve or redesign business processes involves the adaptation of the underlying information systems.

The relationship between BPM and Process Mining is depicted in Figure 1.3. It presents the BPM life cycle, subdivided in four primary phases: Design, System Configuration, Execution and Diagnosis. As such it shows how operational business processes are conceived, set up, enacted and analyzed. Obviously, Process Mining is situated in the diagnosis phase of the BPM life cycle. The start of the life cycle is the design phase. In many organizations, a priori models of business processes are developed as they represent how the business process are assumed or expected to be executed. Process modeling is the key technical subphase of design where informal descriptions are formalized using a specific business process modeling language.

The next step is to translate those models into an operating process-aware information system (PAIS), represented by the second BPM life cycle phase, namely system configuration. Ideally, a dedicated business process management system (BPMS) is used to realize the business process. However, in practice, we often find that a priori models serve as a basis for the configuration of a PAIS, but the process itself is implemented by a set of rules and policies that need to be complied with. As a result, there often exists a divergence between the process design and the implemented system. This difference is denoted by Gap 1 in Figure 1.3.

Process enactment is the third phase in the BPM life cycle. Van der Aalst [132] states that process enactment is the usage of the information system software, configured in a former step to support process execution. An implemented system will inevitably grant a certain level of freedom to its users since total control on the actual execution is often unattainable and unwanted. However, this might lead to situations where people use the system in a different way than envisaged by the systems designers. Furthermore, even without large degrees of freedom, it is often observed that people bypass certain constraints. Such deviations from prescribed behavior by the information systems are termed workarounds. The second gap in Figure 1.3 represents the discrepancy between what the information system prescribes and how processes are actually carried out.

The indicated gaps in the BPM life cycle are the main drivers of the fourth phase, namely diagnosis. When operational routines are carried out, it can be

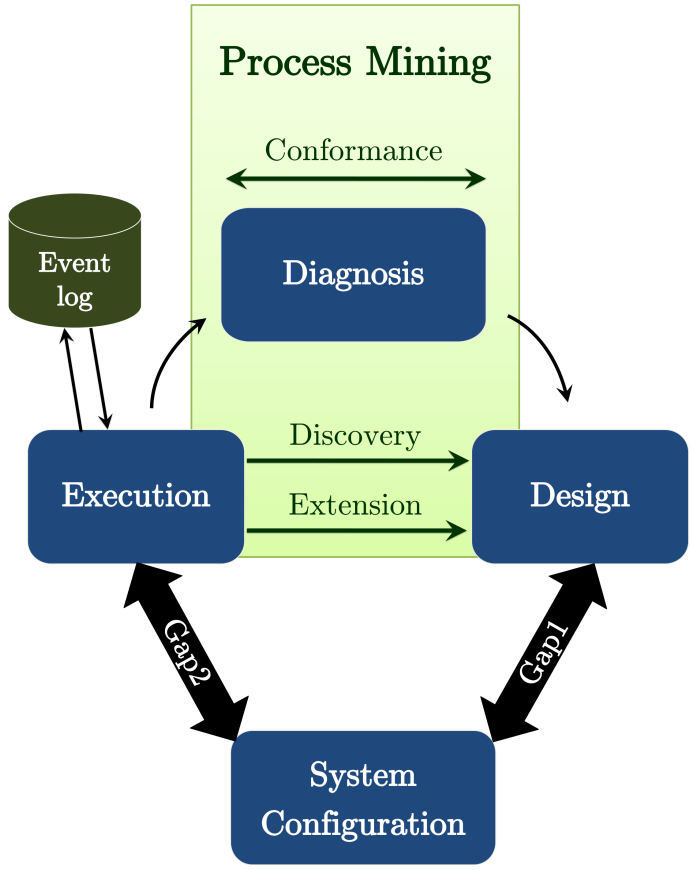


Figure 1.3: Process Mining within the BPM life cycle

assumed that data about each single step in the process are recorded by the information system in some kind of process-oriented data warehouse. As such, process information becomes available for profound analysis. Process Mining is an entire part of the diagnosis phase in the BPM life cycle. In contrast to Business Process Intelligence (BPI) and Business Activity Monitoring (BAM), domains that also fit in this phase, Process Mining is a more powerful set of methods to deal with a thorough, bottom-up investigation. Process mining techniques do not assume any of the critical points to be known upfront since this type of analysis will often start with the discovery of the actual business processes from the recorded data. The exploratory nature of Process Mining allows a relatively unbiased examination of the business process at hand. In this way, Process Mining proves to be an ideal means for guiding process improvement and redesign approaches.

1.3 Knowledge Discovery in Databases

As well as with the field of BPM, Process Mining is tightly coupled with Knowledge Discovery in Databases (KDD). In fact, Process Mining can be situated at the intersection of both fields, as shown in Figure 1.4. Consequently, in this section, the common grounds with the field of KDD will be outlined.

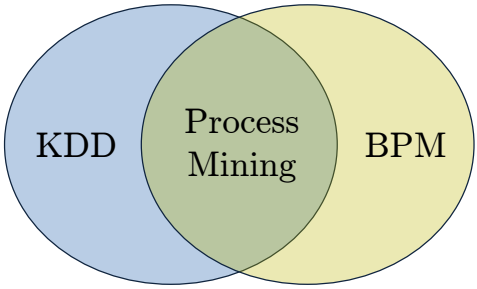


Figure 1.4: Process Mining at the intersection of the fields of KDD and BPM

Fayyad et al. [52] describe the domain of Knowledge Discovery in Databases (KDD) as follows: the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data. Data mining techniques play an important role in this process since they actually extract patterns from data. In the last decade, the importance of KDD and data mining has grown significantly since organizations that excel with respect to their analytical capabilities create a significant competitive advantage [37]. In order to situate the most important process mining learning tasks, two important distinctions in the KDD-field are presented: descriptive vs. predictive learning and attribute-value vs. relational learning.

1.3.1 Descriptive vs. Predictive Learning

A first traditional distinction involves the two high level, primary goals of data mining: prediction and description. Predictive data mining employs supervised learning algorithms in order to predict a target variable that is known upfront. Predictive data mining can be categorized into classification tasks and regression tasks. Classification entails a discrete target variable while a continuous target variable is used for regression. Typical descriptive data mining techniques are clustering and association rule mining. Here, the objective is to derive patterns that summarize the underlying relationships in data. This type of data mining tasks is often denoted as unsupervised learning since there is no target variable known upfront.

1.3.2 Attribute-Value vs. Relational Learning

Another important issue concerns the data itself from which patterns are to be discovered. Within an attribute-value context, the data can be summarized in one table, where each row represents one example and each column corresponds to one feature. A vast majority of learning algorithms use an attribute-value representation of the data. Though these techniques are very successful and efficient, many problems and data sets exist for which this limited expressivity is an important drawback. Accordingly, learning frameworks have been developed that approach this limited expressivity. Inductive logic programming (ILP) [95], also called relational learning or first order learning, is a research field that studies learning within the representations offered by logic and logic programming. Its primary aim is to develop machine learning methods that are able to cope with structured data in the form of entities and relationships.

1.3.3 Specificities of Process Mining Tasks

The limited expressivity of traditional attribute-value data mining algorithms proves also problematic for application to the crucial learning task in Process Mining, i.e. process discovery. Process discovery is an inherently descriptive learning task since the goal is to discover a process model from a set of process executions captured in the event log. As such, the hypothesis space of this learning problem is very specific an highly complex because of the semantics that underly typical business process modeling notations. One approach is to take advantage of techniques in the domain of sequence learning. For instance, the use of Hidden Markov Models [104] or Conditional Random Fields [80] can be expected to be useful for process discovery. However, such models lack expressive power as well in order to represent the underlying semantics of a process model.

Due to the complexity of the hypothesis space and the limited or inappropriate expressivity of existing learning techniques, different proprietary algorithms have been developed for process discovery. However, some of them employ ideas from traditional learning frameworks. For instance, Genetic Miner [10] takes advantage of the ideas in the field of evolutionary computation to induce a process model from a set of process executions. Furthermore, Goedertier et al. [63] demonstrate that process discovery can be translated to a first order (relational) classification problem with the use of artificial negative events.

Although the main learning task in Process Mining proves especially difficult with respect to existing learning approaches, there is of course room for the application of regular data mining algorithms to other process mining tasks. Especially enhancement tasks are subject of these advanced analysis techniques. Example studies are [113], [121] and [140]. In conclusion, Process Mining has lots of common grounds with Knowledge Discovery in Databases since these domains share a similar goal of finding interesting patterns in large amounts of data. It was shown that process discovery is a somewhat peculiar learning task since it is difficult to make use of traditional learning approaches, mainly because of their lack of expressive power.

1.4 Structure and Contributions

This section outlines the main theoretical and practical contributions of this text.

1.4.1 Metrics for Quantifying the Quality of Discovered Process Models

The first part of this thesis covers the problem of evaluating the quality of discovered process models. The foundation of this research area is the study on conformance checking by Rozinat and van der Aalst [107]. The quality of a discovered process model can be judged along a multitude of different perspectives. These perspectives can be categorized into two high-level dimensions: accuracy and comprehensibility.

Our contribution is twofold. First of all, an evaluation study is performed on existing evaluation metrics. With this study, the advantages and drawbacks of currently available metrics are mapped. Secondly, based on the idea of inducing artificial negative events into an event log [63], a novel precision metric is proposed. This precision metric serves as the input for the application of the well-known F-score for process discovery evaluation. This novel evaluation methodology is considered an important step towards the definition of a more comprehensive evaluation framework for discovered process models.

These contributions are described and published in:

- J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens. **A Critical Evaluation Study of Model-Log Metrics in Process Discovery**. In M. zur Muehlen and J. Su, editors, *Business Process Management Workshops*, 66 of *Lecture Notes in Business Information Processing*, pages 158–169. Springer, 2010.

- J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens. **A robust F-measure for evaluating discovered process models**. In *CIDM*, pages 148–155. IEEE, 2011.

1.4.2 A Multi-Dimensional Quality Assessment of Process Discovery Techniques

Within the field of Process Mining, there is a traditional reliance on artificial data in order to develop learning algorithms. This focus on artificially generated data is logical because of two reasons. Firstly, it allows for researchers to develop algorithms that are able to mine special process constructs. Secondly, relying on artificial data allows for a straightforward analysis of the correctness of a process discovery technique since the behavior that is part of the artificially generated event log is known upfront. Accordingly, the evaluation of the quality of process discovery techniques in practice has received only modest attention. In this respect, this work contributes to the literature with the first multi-dimensional quality assessment of state-of-the-art process discovery techniques using eight real-life event logs. Furthermore, the discovery techniques gauged along three key quality criteria, namely accuracy, comprehensibility and scalability. This study is published in:

- J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens. **A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs**. *Information Systems*, 37(7):654–676, 2012.

1.4.3 Improving Process Discovery with Active Trace Clustering

By far the most arduous challenge for process discovery algorithms consists of tackling the problem of accurate and comprehensible knowledge discovery within highly flexible business process environments. Event logs from such flexible systems often contain a large variety of process executions which makes the application of Process Mining most interesting. However, simply applying existing process discovery techniques will often yield highly incomprehensible process models

because of their inaccuracy and complexity. With respect to resolving this problem, trace clustering is one very interesting approach since it allows to split up an existing event log so as to facilitate the knowledge discovery process. In this chapter, a novel trace clustering technique is described which significantly differs from previous approaches. Above all, it starts from the observation that currently available techniques suffer from a large divergence between the clustering bias and the evaluation bias. By employing an active learning inspired approach, this bias divergence is solved. In an assessment using both a controlled environment as well as four real-life event logs, it is shown that our technique significantly outperforms currently available trace clustering techniques from a process discovery evaluation perspective.

Chapter 4 has been submitted for publication in:

- J. De Weerdt, S. vanden Broucke, J. Vanthienen, and B. Baesens. **Active Trace Clustering for Improved Process Discovery**. *Submitted to IEEE Transactions on Knowledge and Data Engineering*, 2012.

1.4.4 Real-Life Case Studies

The findings described in the previous chapters are further validated with a number of real-life case studies.

These applications are published in:

- J. De Weerdt, A. Schupp, A. Vanderloock, and B. Baesens. **Process Mining for the multi-faceted analysis of business processes - A case study in a financial services organization**. *Computers in Industry*, Forthcoming.

- S. Goedertier, J. De Weerdt, D. Martens, J. Vanthienen, and B. Baesens. **Process discovery in event logs: An application in the telecom industry**. *Appl. Soft Comput.*, 11(2):1697–1710, 2011.

- J. De Weerdt, F. Caron, J. Vanthienen, and B. Baesens. **Getting a Grasp on Clinical Pathway Data: An Approach Based on Process Mining**. In *The Third Workshop on Data Mining for Healthcare Management at the 16th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Kuala Lumpur, Malaysia (forthcoming)*, 2012.

- J. De Weerdt, S. vanden Broucke, J. Vanthienen, and B. Baesens. **Leveraging process discovery with trace clustering and text mining for intelligent analysis of incident management processes**. In *IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2012.

Chapter 2

Evaluating Discovered Process Models

One of the essential issues in both the development and application of process discovery techniques is the availability of objective metrics that are capable of quantifying the quality of a discovered process model. The usefulness of process discovery evaluation metrics is determined by the following requirements:

Construct validity. The extent to which a metric, being the operationalization of a certain evaluation dimension (e.g. recall, precision, etc.), does actually measure the quality of a discovered process model in respect to the theoretical definition of this evaluation dimension.

Reproducibility. The extent to which a process discovery evaluation metric consistently and repeatedly presents identical (or very similar results) for identical data and model input combinations.

Scalability. The extent to which the calculation of a metric is computationally demanding.

Understandability. The extent to which a metric is comprehensible to both process analysts as well as business stakeholders.

2.1 The Broader Area of Conformance Checking

In this chapter, we adopt a distinct focus on the evaluation of discovered process models. Nonetheless, this area of research is part of a much broader domain which is called conformance checking. Next to evaluation metrics for discovered process

models, techniques proposed in this area can also be applied to existing or designed models where they can be used to inspect the level of compliance between a model and a log, i.e. compliance analysis. In this way, the application domain of conformance checking techniques is much broader, with demonstrated applications in auditing [143], business process forensics [125], performance analysis [136] and prediction [140].

However, by far the most relevant evaluation dimension for compliance analysis is the recall or fitness perspective. This is because one is mainly interested in finding these deviations where observed behavior does not conform with prescribed behavior in the model. In contrast, for the evaluation of discovered process models, it is always required to take a multi-perspective view. As such, this chapter does not present a comprehensive analysis of the broader domain of conformance checking. This is also the reason why techniques such as footprint comparisons [135] and compliance analysis based on behavioral profiles [151] are not discussed. It is pointed out however that the metrics presented in [151] can be used for analysis of discovered process models. However, as with footprint-based metrics, these techniques often suffer from their global, rather coarse-grained nature when applied to the evaluation of discovered process models.

2.2 Main Evaluation Dimensions for Discovered Process Models

The quality of a discovered process model can be judged along a multitude of different perspectives. These perspectives can be categorized into two high-level dimensions: accuracy and comprehensibility. A characterization of the quality evaluation setting of discovered process models is presented in Figure 2.1.

Evaluation trade-offs. The accuracy evaluation of a discovered process model should ideally consist of three perspectives: recall, precision and generalization. Good recall is an imperative requirement for any discovered process model because it signifies how much behavior present in the event log is captured by the model. Next to recall, models also need to be precise. Accordingly, precision metrics quantify whether a process model is too general with respect to the behavior in the event log. Finally, generalization metrics punish process models that overspecify the behavior in the event log. The key challenge for any process discovery technique is to find the right trade-off between recall, precision and generalization requirements. In some situations, recall will be a focal point while in another setting, finding a right balance between a model that is precise enough while still allowing for behavior that is not encountered in the data is required. Because of these considerations, a well-defined evaluation framework is a key element in

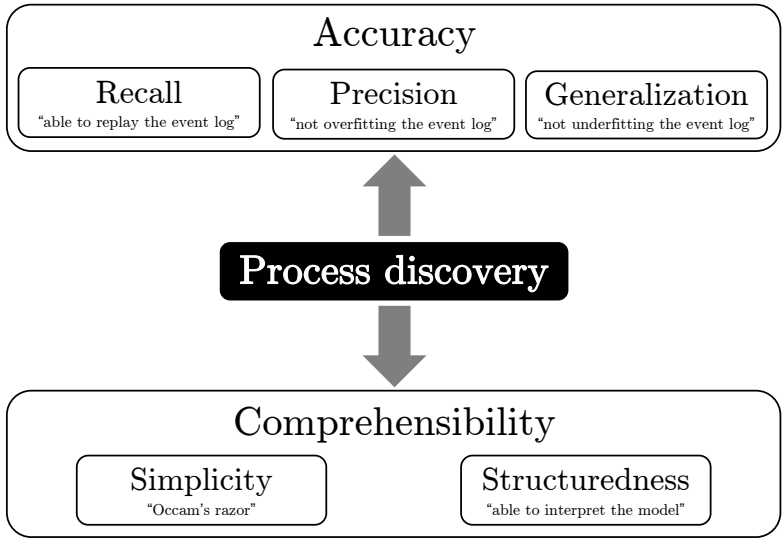


Figure 2.1: Quality dimensions for evaluating discovered process models

further developing and improving process discovery techniques themselves.

To this, it should be added that accuracy evaluation is not sufficient for deciding upon the quality of a discovered process model. In a similar way as comprehensibility being an important issue in the data mining domain, also for process discovery, simplicity and structuredness of a learned model are essential elements that determine the usefulness of a learning technique. For discovered process models, we identify two components of process model comprehensibility, albeit both are definitely correlated. Firstly, simplicity denotes the principle that "all other things equal, a simpler explanation is better than a more complex one". In the field of machine learning, this principle is famously known as Occam's razor. As to discovered process models, this principle is mainly embodied by the number of control-flow constructs contained in the model. Furthermore, the complexity of the used control-flow patterns can also influence simplicity. Further, structuredness is another component of comprehensibility. In contrast to simplicity, structuredness is more about the characteristics and ease of interpretation of the modeling notation. As such, the representation of a discovered process model is often a key determinant of its comprehensibility.

In the remainder of this chapter, the focus will be set on the evaluation of the accuracy of discovered process models. Hereto, an overview of currently available metrics is presented which is followed by a discussion on how the F-measure can contribute to the development of a proper process discovery evaluation framework.

2.3 An Overview of Accuracy Evaluation Metrics

For the evaluation of mined process models, appropriate metrics need to be at hand that quantify the different evaluation dimensions. It is pointed out that multiple evaluation methodologies can be identified [108]. One method is to compare the traces in the event log and the model mined from this event log. In the remainder of this chapter, such a metric is called a *model-log metric*. Despite the fact that model-log metrics are a highly suitable quantification approach, there exist other ways for discovered process model evaluation. An alternative for model-log metrics are metrics that quantify the difference between two process models [8]. This type of metrics can be used to compare an existing process model with a discovered model. However, it should be noted that these metrics cannot be employed for this evaluation study because we do not have any ex-ante process models. Finally, two totally different evaluation approaches are available that have their origins in the field of data mining. Firstly, Rozinat et al. show in [108] how Hidden Markov Models (HMMs) can be employed for process discovery evaluation. Furthermore, Calders et al. [25] propose to evaluate the quality of mined process models by making use of the minimum description length (MDL) principle. Although these are interesting approaches, their actual applicability is currently limited and are therefore left out of this analysis.

2.3.1 Currently Available Model-Log Metrics

As for model-log metrics, Cook and Wolf [34] are to be considered the first researchers to quantify the relationship between process models and event logs. In the context of software processes, they compare event streams from both the model and the log by making use of different string distance metrics. Table 2.1 provides an overview of the most important model-log metrics in the process mining domain. The table shows that currently available metrics evaluate discovered process models along three important dimensions. As such, it can be seen that a majority of evaluation metrics reflect the recall perspective. However, good recall is not the only dimension a model should score well on. The usefulness of models is also determined by a good balance between precision and generalization.

The recall dimension. Recall or sensitivity is the primordial accuracy evaluation perspective. This dimension reflects how much behavior present in the event log is captured by the model. For every process discovery algorithm, it is of utmost importance to render models with good recall because representing the control-flow behavior in an event log is the major objective of any technique. Hence, it is definitely satisfying that a number of researchers have proposed different measures to capture recall.

Name	Symbol	Author	Available in ProM	Range	Model input type	Perspective		
						Recall	Precision	Generalization
Continuous Parsing Measure	*CPM*	Weijters et al. [153]	✓	[0,1]	Heuristic net	✓		
Parsing Measure	*PM*	"	✓	[0,1]	Heuristic net	✓		
Completeness		Greco et al. [66]		[0,1]	Workflow schema	✓		
Soundness		"		[0,1]	Workflow schema		✓	
Fitness	f	Rozinat and van der Aalst [107]	✓	[0,1]	Petri net	✓		
Behavioral appropriateness	a_B	"	✓	[0,1]	Petri net		✓	
Advanced behavioral appropriateness	$a_B^{'}$	"	✓	[0,1]	Petri net		✓	
Structural appropriateness	a_S	"	✓	[0,1]	Petri net			✓
Advanced structural appropriateness	$a_S^{'}$	"	✓	[0,1]	Petri net			✓
Partial fitness - complete	$PF_{complete}$	Alves de Medeiros et al. [7]	✓	[-∞,1]	Heuristic net	✓		
Behavioral recall	r_B^p	Goedertier et al. [63]		[0,1]	Petri net	✓		
Behavioral specificity	s_B^n	"		[0,1]	Petri net		✓	
ETC precision	etc_P	Muñoz-Gama et al. [96]	✓	[0,1]	Petri net		✓	
Average alignment-based trace fitness	f_a^{avg}	van der Aalst et al. [136]	✓	[0,1]	Petri net	✓		
One align precision	a_p^1	Adriansyah et al. [1]	✓	[0,1]	Petri net		✓	
Best align precision	a_p	"	✓	[0,1]	Petri net		✓	

Table 2.1: Overview of process mining evaluation metrics: model-log metrics

- *Fitness* (f) is a metric that is obtained by trying whether each trace in the event log can be reproduced by the generative model, in this case a Petri net. This procedure is called sequence replay [107]. During replay, the transitions in the Petri net will produce and consume tokens to reflect the state transitions. Consequently, the fitness measure punishes for tokens that must be created additionally in the marked Petri net and also for tokens that remain after replay.

- The *Continuous Parsing Measure* (CPM) was proposed by Weijters et al. [153]. This metric is very similar to fitness (f) from [107], nonetheless CPM is a recall metric for heuristic nets. As such, although the idea behind the metric is similar in terms of the incorporation of missing and remaining tokens collected during trace replay, the implementation and the actual metric definition are slightly different.

- The *Parsing Measure* (PM) was also proposed by Weijters et al. [153]. It quantifies the percentage of traces in the log that can be replayed by the discovered process model. It should be noted that PM is a coarse-grained metric. A single missing arc in the discovered process model can result in parsing failure for all traces.

- A very similar metric is *completeness* as defined by Greco et al. [66]. This is the percentage of traces in the event log that are compliant with the workflow schema or process model. Completeness always ranges between 0 and 1.

- *Partial fitness - complete* ($PF_{complete}$) is a metric that was proposed by Alves de Medeiros et al. [10] in the context of the development of the Genetic Miner algorithm. This metric is very similar to the fitness metric (f), but it additionally exploits trace frequencies in order to take into account the severity of missing and remaining tokens. This metric corresponds to the *Improved Continuous Semantics fitness* (ICS), which is employed in Chapter 4. Note that the *partial fitness - precise* ($PF_{precise}$) proposed in the same study cannot be considered a model-log metric because it takes into account an entire set of process models.

- *Behavioral recall* (r_B^p) as defined by Goedertier et al. [63] is the percentage of correctly classified positive events in the event log. During sequence replay, it is verified whether every positive event can be parsed by the model. Note that this measure originates from a process discovery technique that makes use of inducing artificial negative events in order to mine control-flow models. By verifying the parsing of positive events, recall can be quantified.

- *Average alignment-based trace fitness* (f_a^{avg}) is the most recent recall metric in the process mining domain [136]. In contrast to a large majority of recall metrics, the metric is based on aligning a process model and an event log [2, 3, 4] instead of replaying the log in the model. As such, the metric punishes for alignments which require model (log) moves without a corresponding move in the log (model).

The precision dimension. The trade-off between precision and generalization is a major challenge in process discovery. Although models should be precise, generalizing beyond observed behavior is also a necessity. This is because assuming that all behavior is included in an event log is a much too strong completeness assumption. So, process discovery algorithms should be able to balance between underfitting (overly general models) and overfitting (overly precise models). Therefore, superior precision and generalization metrics should be at hand. The following list enumerates the currently available measures.

- With the definition of *soundness*, Greco et al. [66] were first to define a precision metric for evaluating a discovered process model. Soundness is defined as the percentage of traces compliant with the process model that have been registered in the log. Calculating soundness is not straightforward because enumerating all possible paths in a process model is hard. Even for smaller process models, it might be impossible to determine all the traces that are compliant with a process model.

- In [107], Rozinat and van der Aalst defined four different appropriateness measures. The simple *behavioral appropriateness* (a_B) measures the amount of possible behavior to determine a mean number of enabled transitions during log replay. Because this metric depends on structural properties of the process model, it is advised to use the advanced behavioral appropriateness.

- *Advanced behavioral appropriateness* ($a_B^{'}$) allows to compare the behavior that is specified by the model with the behavior that is actually needed to describe the behavior in the event log. Therefore, this metric makes use of an analysis of "follows" and "precedes" relations, both in the model and the event log. Comparing the variability of these relations allows the definition of a precision metric that penalizes extra behavior.

- *Behavioral specificity* (s_B^n) is the percentage of correctly classified negative events during sequence replay. As such, it is the counterpart of behavioral recall (r_B^p). Artificial negative events can be generated with the technique developed by Goedertier et al. [63]. However, the definition of the metric does not exclude the use of natural negative events or negative events stemming from other techniques.

- *ETC precision* (etc_P) is a more recently proposed precision metric [96, 97]. This metric is based on the construction of a prefix automaton for the event log at hand. By taking into account the number of so-called escaping edges while mapping the behavior in the event log with the behavior in the model, a state-of-the-art precision metric is defined. Note that this metric is implemented as a plug-in in ProM 6.

- *One align precision* (a_p^1) and *best align precision* (a_p) extend the etc_P metric by first aligning the log and the model [1]. In this way, the main problem of etc_P, i.e. the fact that precision is assessed as long as the trace under investigation remains fitting, is solved. Both metrics have been implemented in ProM.

The generalization dimension. The third and final accuracy evaluation perspective is generalization. Metrics quantifying this dimension should punish process models which are overly precise, thus not allowing unseen but very likely behavior when taking into account the data in the event log. In the process mining literature, no true generalization metrics for discovered process model have been proposed. Nevertheless, the structural appropriateness metrics from [107] adhere most closely to the concept of evaluating the generalization capability of a discovered process model.

- *Structural appropriateness* (a_S) is based on the number of different task labels in relation to the graph size of the model. As identified by Rozinat and van der Aalst [107], this metric's applicability is limited because it is only based on the graph size of the model.

- *Advanced structural appropriateness* ($a_S^{'}$) is a generalization metric that evaluates two specific design guidelines for expressing behavioral patterns. This measure will punish for both alternative duplicate tasks and redundant invisible tasks. Note that these guidelines are definitely not the only behavioral preferences of control-flow models. However, $a_S^{'}$ is the only metric that quantifies generalization in some way. Ideally, a process model does not contain redundant invisible tasks nor alternative duplicate tasks. Accordingly, this measure will punish for models that simply enumerate all traces in the event log and for models that entail to much irrelevant invisible tasks. Strictly speaking, only alternative duplicate tasks are a true indicator for overspecialization, while irrelevant invisible tasks are an indicator of a lack of model simplicity.

In conclusion, three important dimensions were identified along which process models should be judged: recall, precision and generalization. Many evaluation

metrics have already been proposed in literature with a majority of these metrics judging the recall of a discovered process model. Nevertheless, the trade-off between underfitting and overfitting models is crucial as well. Hereto, currently available precision and generalization metrics might be insufficient so as to capture these underlying evaluation dimensions comprehensively and correctly. This issue will be investigated in the following section.

2.3.2 Elucidating Key Evaluation Metrics in Process Discovery

In this section, the most important currently available process mining metrics are illustrated so as to show how they capture the different dimensions discussed in the previous section. Five metrics were selected: fitness, advanced behavioral appropriateness, advanced structural appropriateness, behavioral recall and behavioral specificity, so that the three identified dimensions are covered by at least one metric. Although the other metrics definitely have value, they do not add to this analysis because most of them are very comparable to one of the metrics selected. Furthermore, metrics like soundness and both simple appropriateness measures suffer from different shortcomings and are therefore left out.

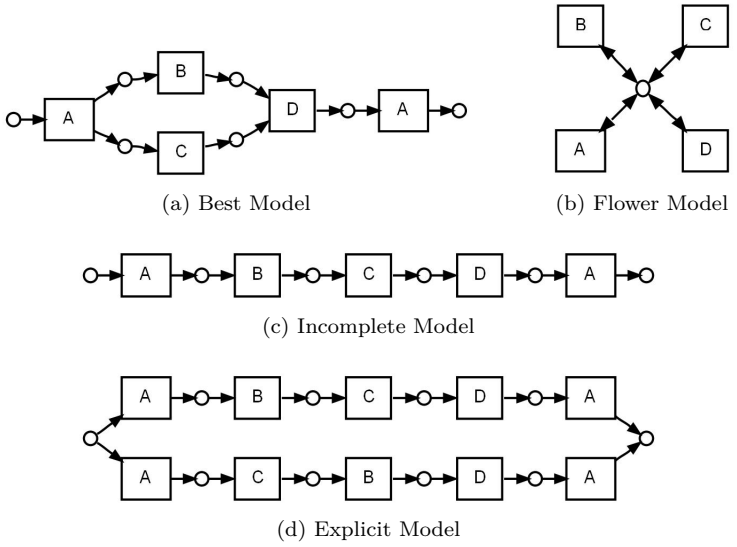
A simplified example to elucidate the most important metrics

The event log of this simplified example contains 50 traces "ABCDA" and 50 traces "ACBDA". Accordingly, the best model representing the traces in the log is model 2.2a. Table 2.2 shows that this model scores 1 on every metric and thus can be considered excellent along all evaluation dimensions.

Metric	**Fitness**	**Adv. Behavioral Appropriateness**	**Adv. Structural Appropriateness**	**Behavioral Recall**	**Behavioral Specificity**
Symbol	f	a'_B	a'_S	r^p_B	s^n_B
(*dimension*)	(*recall*)	(*precision*)	(*generalization*)	(*recall*)	(*precision*)
Best Model	1	1	1	1	1
Flower Model	1	0.17	1	1	0
Incomplete Model	0.92	1	1	0.90	0.89
Explicit Model	1	1	0.40	1	1

Table 2.2: Model-log metrics for a simple artificial event log and four different models

The other models in Figure 2.2 are in one way or the other erroneous. The incomplete model does not recall all the traces in the log, the flower model allows



(a) Best Model

(b) Flower Model

(c) Incomplete Model

(d) Explicit Model

Figure 2.2: Different control-flow models for the simple event log

too much behavior and the explicit model is only a mere enumeration of the traces in the log. We will now elucidate how different metrics are able to identify the dimension(s) along which a mined process model misses the mark.

Fitness. Model 2.2c is not able to capture all the behavior that is present in the event log. Thus, metrics quantifying the recall dimension should indicate this problem. The fitness measure (f) is a particularly useful metric to do so. With 50 traces (n_i) for each of the grouped traces, both the number of missing tokens (m_i) and the number of remaining tokens (r_i) amounts to 1 for the second grouped trace ("ACBDA") and 0 for the first grouped trace ("ABCDA"). Furthermore each of the traces requires 6 tokens to be consumed (c_i) / produced (p_i) in order to be replayed. Accordingly, the fitness of the incomplete model sums up to 0.92. Note that i is an index running over the number of different grouped traces (k), which is two in this simplified case.

- Fitness:

$$f = \frac{1}{2}\left(1 - \frac{\sum_{i=1}^{k} n_i m_i}{\sum_{i=1}^{k} n_i c_i}\right) + \frac{1}{2}\left(1 - \frac{\sum_{i=1}^{k} n_i r_i}{\sum_{i=1}^{k} n_i p_i}\right) \tag{2.1}$$

$$= \frac{1}{2}\left(1 - \frac{50*0 + 50*1}{50*6 + 50*6}\right) + \frac{1}{2}\left(1 - \frac{50*0 + 50*1}{50*6 + 50*6}\right) = 0.92$$

The other models in figure 2.2 are able to reproduce all the behavior in the event log. Accordingly, this is also demonstrated by the fitness measure. During sequence replay, there are no missing tokens nor remaining tokens for any of the other models and thus the fitness evaluates to 1. Notice that a fitness value of 0.92 for the incomplete model is an unattractively high value for a model that only parses half of the traces correctly.

Advanced behavioral appropriateness. The flower model (2.2b) is a generic model that allows any sequence of activities. Because it overgeneralizes, it is useless for any process intelligence activity. Nevertheless, the flower model captures all the behavior in the event log perfectly, so the model is not penalized by a recall metric. The advanced behavioral appropriateness does punish the flower model for its overly general representation.

- Adv. behavioral appropriateness:

$$a_B^{'} = \frac{|S_F^l \cap S_F^m|}{2.|S_F^m|} + \frac{|S_P^l \cap S_P^m|}{2.|S_P^m|} = \frac{2}{2 * 12} + \frac{2}{2 * 12} = 0.17 \qquad (2.2)$$

The advanced behavioral appropriateness for the simplified example flower model is found by calculating the elements in the "sometimes follows" and "sometimes precedes" relations in the model (S_F^m and S_P^m) and in the log (S_F^l and S_P^l) [107]. The calculation for the follows relations is illustrated in figure 2.3.[1] According to ProM, there are a total of 12 sometimes follows relations and 12 sometimes precedes relations in the model. However, the model and the log have only 2 sometimes follows and 2 sometimes precedes relations in common. Therefore, $a_B^{'}$ adds up to 0.17.

For perfectly precise models, $a_B^{'}$ will equal 1. This is illustrated in table 2.2 as the other three models are not overgeneralizing and their $a_B^{'}$ evaluates to 1 accordingly. For models 2.2a and 2.2d, the sometimes follows and sometimes precedes relations in the model and the log are exactly the same. For model 2.2c, the relations are not completely identical, but the model is in fact more restrictive with respect to the relations in the log, so the generalization measure evaluates to 1. This signifies correctly that the model is not underfitting. Other metrics should indicate the incompleteness of the model.

An important remark should be made concerning the calculation of $a_B^{'}$. The calculation involves a state space analysis which can be computationally very demanding. Furthermore, the calculation seems approximate because a manual analysis of the metric for the flower model results in a total of 16 sometimes follows and sometimes precedes relations. Apparently, (A,A), (B,B), (C,C) and

[1] For clarity, we make abstraction of the artificial Start and End activities.

↙	A	B	C	D
A	$\mathbf{A_F}$	S_F	S_F	S_F
B	S_F	$\mathbf{A_F}$	S_F	S_F
C	S_F	S_F	$\mathbf{A_F}$	S_F
D	S_F	S_F	S_F	$\mathbf{A_F}$

↙	A	B	C	D
A	$\mathbf{S_F}$	S_F	S_F	S_F
B	S_F	$\mathbf{S_F}$	S_F	S_F
C	S_F	S_F	$\mathbf{S_F}$	S_F
D	S_F	S_F	S_F	$\mathbf{S_F}$

(a) Model relations: calculated by ProM (l) and manually (r)

↙	A	B	C	D
A	A_F	A_F	A_F	A_F
B	A_F	N_F	S_F	A_F
C	A_F	S_F	N_F	A_F
D	A_F	N_F	N_F	N_F

(b) Log relations

Figure 2.3: Follows relations for the flower model (2.2b) and for the simplified event log

(D,D) relations are categorized as always follows/precedes (see figure 2.3a), despite the fact that the flower model allows more variability for these relations. As such, $a_B^{'}$, as obtained from ProM, slightly underestimates the overgenerality of the flower model.

Advanced structural appropriateness. Another problem in process discovery are overly precise models or overfitting models. Figure 2.2d shows an explicit model that is a mere enumeration of the traces in the log. Again, such a model is undesired and should be punished by a generalization measure. Advanced structural appropriateness is the only currently available metric that allows to quantify some kind of generalization, albeit this metric is not entirely in line with the concept of generalization. Because model 2.2d contains six alternative duplicate tasks (T_{DA}) and no redundant invisible tasks (T_{IR}), $a_S^{'}$ evaluates to 0.4 (note that $|T|$ denotes the total number of tasks). These alternative duplicate tasks are activities B, C and D, occurring once in each of the branches of the explicit process model. They are alternative duplicate tasks because they never happen together in one execution sequence.

- Structural appropriateness:

$$a_S^{'} = \frac{|T| - (|T_{DA}| + |T_{IR}|)}{|T|} = \frac{10 - (6 + 0)}{10} = 0.40 \tag{2.3}$$

The occurrence of alternative duplicate tasks signposts overly specific process models. Nonetheless, this is far from the only determinant of a lack of generalization capabilities. In the process mining literature, there are no metrics available that are able to quantify generalization comprehensively. The problem with the definition of a good generalization metric lies in the fact that the process model needs to be explored in order to investigate the allowance of likely but unobserved process instances. This exploration can quickly become computationally intractable.

Artificially generated negative event metrics. In [63], Goedertier et al. propose two state-of-the-art metrics originating from a process discovery technique called AGNEs (Artificially Generated Negative Events). This technique involves the induction of negative events in the event log in order to allow the application of advanced machine learning techniques (Inductive Logic Programming) for control-flow discovery (see also [56]). The availability of both positive and artificial negative events allows the definition of behavioral recall r_B^p and behavioral specificity s_B^n, metrics that are grounded in traditional data mining theory. According to their definition, they are able to penalize inaccurate process models.

The induction procedure generates artificial negative events as displayed in Table 2.3. The induction technique will be detailed in Section 2.4. A total of 28 negative events are induced into the event log. In order to calculate the metrics, both the positive and negative events are evaluated with a trace replay procedure. Such a trace replay procedure will classify each positive and artificial negative event and as such one can construct a confusion matrix. The confusion matrix for the incomplete process model (2.2c) is shown in Table 2.4. Note that the negative events denoted in bold face in Table 2.3 are incorrectly parsed negative events for model 2.2c in the sense that during replay, these negative events are evaluated as enabled considering the marking of the Petri net after executing the positive events up till their respective positions.

	Trace 1					**Trace 2**				
Positive events	A	B	C	D	A	A	**C**	B	D	A
Artificially	B^n	A^n	A^n	A^n	B^n	B^n	A^n	A^n	A^n	B^n
generated	C^n	D^n	B^n	B^n	C^n	C^n	D^n	C^n	B^n	$\mathbf{C^n}$
negative events	D^n		D^n	C^n	D^n	D^n		$\mathbf{D^n}$	$\mathbf{C^n}$	D^n

Table 2.3: Artificially generated negative events for the simplified event log

Behavioral recall and behavioral specificity for the incomplete model are calculated according to the following formulae.

	true pos.	true neg.	total
pred. pos.	9	3	12
pred. neg.	1	25	26
total	10	28	

Table 2.4: Confusion matrix as used for calculation of the negative event metrics for the incomplete process model

- Behavioral recall:

$$r_B^p = \frac{\sum_{i=1}^{k} n_i TP_i}{\sum_{i=1}^{k} n_i TP_i + \sum_{i=1}^{k} n_i FN_i} \tag{2.4}$$

$$= \frac{50 * 5 + 50 * 4}{(50 * 5 + 50 * 4) + (50 * 0 + 50 * 1)}$$

$$= \frac{9}{10} = 0.90$$

- Behavioral specificity:

$$s_B^n = \frac{\sum_{i=1}^{k} n_i TN_i}{\sum_{i=1}^{k} n_i TN_i + \sum_{i=1}^{k} n_i FP_i} \tag{2.5}$$

$$= \frac{50 * 14 + 50 * 11}{(50 * 14 + 50 * 11) + (50 * 0 + 50 * 3)}$$

$$= \frac{25}{28} = 0.89$$

It is pointed out that negative event-based evaluation metrics suffer from drawbacks as well. Most importantly, these metrics rely on a trace replay procedure. Because it is computationally intractable to make use of backtracking in order to explore the full state space of a process model, it is required to make use of heuristics. In case there exist process instances that do not fit the process model, trace replay often makes use of the concept of force firing. Force firing entails the firing of a transition in a Petri net that is not enabled. In some situations, force firing creates unwanted side-effects which will influence the classification of artificial negative events. In this way, an unwanted correlation might exist between recall and precision.

Conclusion. From this simplified model, it can be concluded that within process discovery research, different evaluation dimensions are covered by existing model-log metrics. With the purpose of tracking down more shortcomings of available process discovery evaluation metrics, a more extended example is used in the next section.

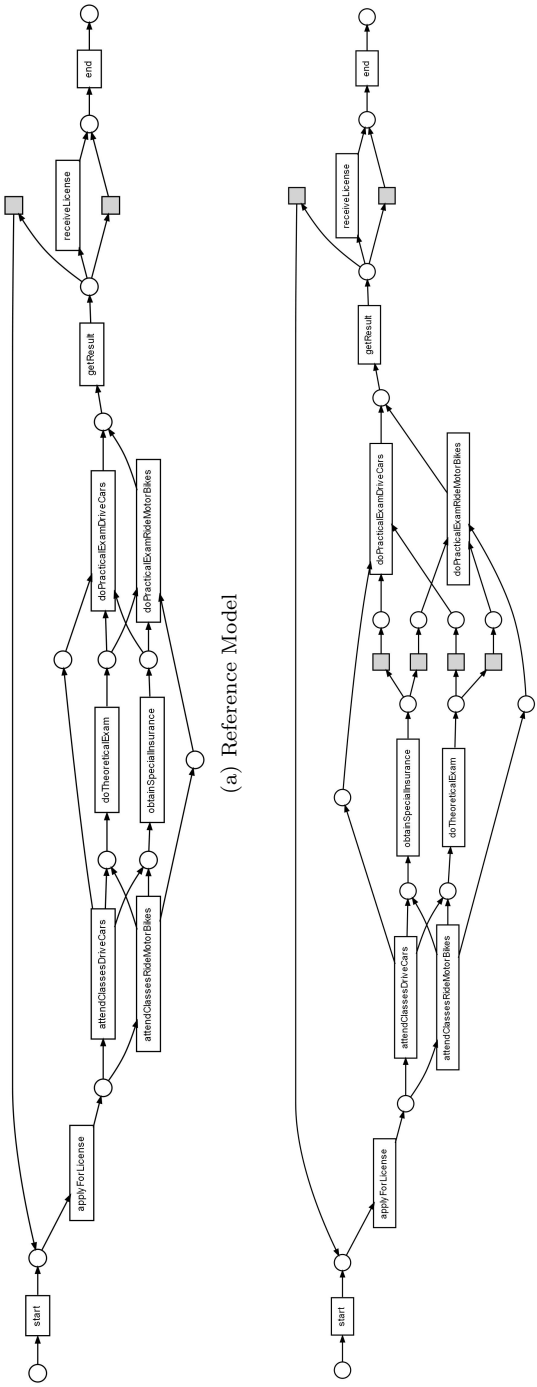
Further insights using an extended Driver's License example

We apply the evaluation metrics to a more extended example by using a modified version of a Driver's License event log (from [7]). This event log and the according models contain more complex control-flow constructs such as a loop and non-free choice constructs. As such, this experiment will allow us to verify whether the available metrics can distinguish between worse and better models in a more complex setting. Figure 2.4 and Table 2.5 show the results of three state-of-the-art process discovery techniques, the reference process model and a flower model. By examining these results, some further elements in the analysis of existing process discovery metrics can be highlighted.

Metric	**Fitness**	**Adv. Behavioral Appropriateness**	**Adv. Structural Appropriateness**	**Behavioral Recall**	**Behavioral Specificity**
Symbol *(dimension)*	f *(recall)*	a'_B *(precision)*	a'_S *(generalization)*	r^p_B *(recall)*	s^n_B *(precision)*
Reference Model	1	0.927	1	1	0.985
Heuristics Model	1	0.874	1	1	0.985
Alpha Model	0.921	1	1	0.917	0.983
AGNEs Model	1	0.906	0.846	1	0.985
Flower Model	1	0.500	1	1	0

Table 2.5: Model-log metrics for the extended Driver's License example

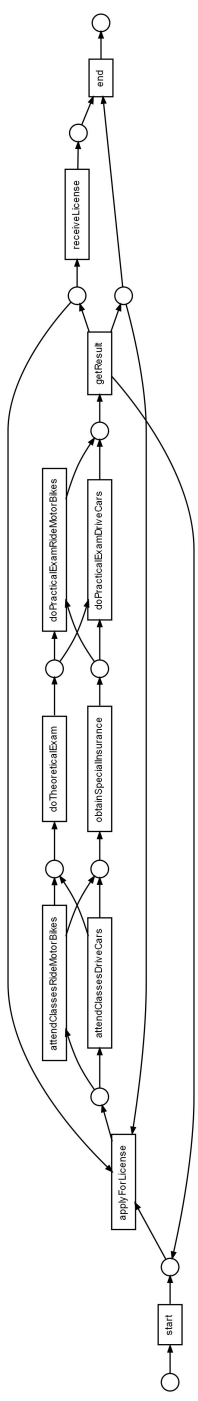
First of all, it can be seen that the fitness metric (f) reveals that every technique, except for the α-algorithm [131], discovers models with perfect recall. This can also be concluded from the behavioral recall metric (r^p_B). Although both metrics seem to measure the same dimension, an important remark should be made. The interpretation of the fitness measure requires some attention: although it accounts for recall as it punishes for the number of missing tokens that had to be created, it also punishes for the number of tokens that remain in the Petri net after log replay. The latter can be considered extra behavior. Therefore, the fitness metric also has a precision semantics attached to it. As mentioned previously, metrics desirably measure only one dimension in order to remain comprehensible.



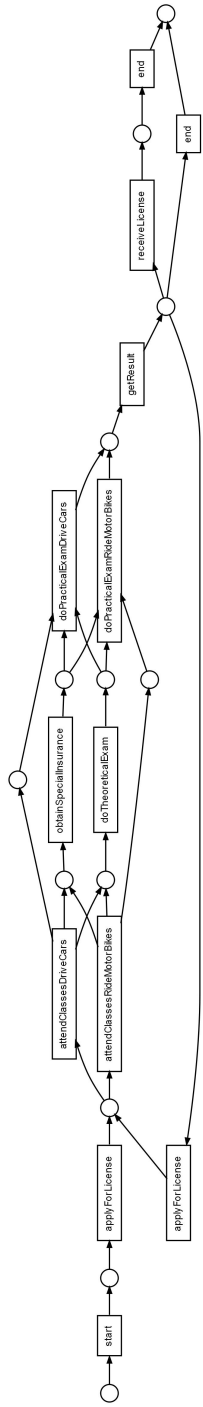
(a) Reference Model

(b) Heuristics Miner result

Figure 2.4: Different control-flow models for the Driver's License event log



(c) Alpha result



(d) AGNEs result

Figure 2.4: Different control-flow models for the Driver's License event log

Secondly, some observations concerning the advanced behavioral appropriateness ($a_B^{'}$) are discussed. As for the flower model in the simplified example, the state space analysis for calculating this metric was calculated swiftly. However, for this Driver's License example, the more or less exhaustive simulation of all the behavior in the model was unable to be completed within an acceptable time period. It can be expected that for even more complex data sets, for instance originating from real-life process environments, the calculation of the metric will be computationally infeasible.

Finally, we notice that it is not obvious which model is to be preferred. Models score differently along distinct dimensions, but there are no rules that define how these dimensions should be put together. Are there dimensions that are more important than others? How can this be included in the analysis phase? How should differences along one and multiple dimensions be assessed? We think that these questions bring about the necessity for a more rigorous and comprehensive evaluation framework for discovered process models.

2.3.3 Observations

Both the explicit illustration of a number of key process discovery evaluation metrics with a simplified example as well as an analysis based on the Driver's License log clarified a number of strengths and weaknesses of currently available metrics. The major shortcomings can be summarized as follows:

- Metrics should be *one-dimensional*. Fitness (f) is an example metric that does not fulfil this requirement. The problem with multi-dimensional metrics is that, unless the different components are clearly identifiable, such metrics will quickly render incomprehensible.

- Many of the currently available precision measures suffer from computational inefficiency. For instance, the advanced behavioral appropriateness metric requires an *exhaustive simulation* of the mined process model. This state space analysis procedure is only *approximate* and this causes difficulties with respect to analyzing discovered process models. Even though conceptually proficient, the imprecise and computationally demanding calculation of the metric brings about the necessity of new precision and generalization measures. The development of better precision metrics is required so as to be able to capture the crucial trade-off between precision and generalization.

- The lack of good quantifiers of the generalization dimension is an important drawback for process discovery evaluation. Advanced structural appropriateness was identified as a metric that captures the lack of the generalization capability of a discovered process model. Nonetheless, this metric is far from

perfect because its original definition strongly focuses on structural aspects of the process model, which only partly determine generalization. Note that generalization metrics are subject of ongoing research. For instance in [136], the authors propose a new approach for quantifying generalization. However, this metric is yet to be implemented.

- Process discovery metrics developed in the light of *machine learning* theory are definitely of added value. Behavioral recall and behavioral specificity are valuable measures, but their integration with the ProM-framework should allow researchers to exploit these state-of-the-art recall and specificity metrics.

Finally, this analysis demonstrates that a more rigorous and comprehensive evaluation framework for process discovery is definitely needed. Although this need has been formulated previously [111], such a framework is still missing. The insights provided can be considered as a valuable impetus hereto. It can be concluded that in any process discovery analysis, combining different metrics is indispensable, as metrics preferably measure only one aspect of a mined process model and models will always have to be judged along multiple dimensions. Therefore, the application of the F-measure for process discovery evaluation is proposed in the next section.

2.4 Towards an Evaluation Framework with the Application of the F-score

Within the research domain of process mining, a lot of attention has been bestowed on process discovery [10, 53, 63, 67, 131, 153, 154, 155]). Many of these algorithms are able to deal with specific problems related to control-flow discovery: parallelism, loops, duplicate tasks, noise, and non-local dependencies. However, the effort spent in developing process discovery algorithms is not counterbalanced by the effort put into the evaluation of the discovered process models. As such, it was identified in the previous section that a well-defined evaluation framework for process discovery is still missing. The lack of an evaluation framework is primarily due to the difficulty of combining metrics that capture different dimensions along which process models should be evaluated. More specifically, process models cannot be evaluated on recall or sensitivity only. Although this dimension is of utmost importance, other requirements for discovered process models such as precision and generalization beyond observed behavior should be included in any process discovery evaluation analysis.

In this section, an evaluation approach is described which is based upon artificially generated negative events. This approach stipulates the application of the

well-known F-measure [147] to discovered process models. With the purpose of applying the F-score, a novel precision metric is proposed in Section 2.4.2 which is similar to the behavioral specificity metric described earlier. With the availability of recall and precision metrics, the new evaluation approach is proposed in Section 2.4.3. The approach is illustrated with a small scale example in Section 2.4.4. Note also that the next chapter will elaborate on the empirical validation of the proposed F-score evaluation methodology with a more extensive evaluation environment for process discovery algorithms.

2.4.1 The Induction of Artificial Negative Events

Event logs rarely contain information about transitions that are not allowed to take place. This makes process discovery an inherently unsupervised learning problem. To make a tradeoff between overly general and overly precise process models, learners make additional assumptions about the given event sequences. Such assumptions are part of the inductive bias of a learner. For instance, process discovery algorithms might include the assumption that event logs portray the complete behavior of the underlying process and implicitly use this strong completeness assumption to make a tradeoff between overly general and overly precise process models. Such a strong completeness assumption is often unwanted because event logs will typically not portray all possible behavior in a certain business process.

In [63], Goedertier et al. describe a technique to artificially generate negative events based on an event log containing only positive events. The induction procedure is the foundation for their process discovery technique called AGNEs Miner. In this section, we will make use of the same principle for generating negative events in order to build our evaluation approach.

Principle

The technique of inducing negative events is relatively straightforward. Negative events record that at a given position in an event sequence, a particular event cannot occur. At each position in each event trace in the log, it is examined which negative events can be recorded for this position. In a first step, the technique stipulates that the event log is made more compact, by grouping process traces that have identical sequences into grouped process instances. By grouping similar process instances, searching for similar behavior in the event log can be performed more efficiently.

In the second step, all negative events are induced for each grouped process instance. Negative examples can be introduced in grouped process instances by checking at any given positive event whether any other activity type in the event log could occur as an event at this position. For each of these events, it is tested

whether there exists a similar sequence in the event log in which at that point the event under consideration occurs. If such an event does not occur in any other sequence, such behavior is not present in the event log. Consequently, a negative event can be added at this position in the event sequence. On the other hand, if a similar sequence is found with this behavior, no negative event is generated.

Example

In order to elucidate the principle of generating artificial negative events more clearly, the injection procedure is illustrated with a small example. Take the event log in Figure 2.5, which is perfectly represented by the process model in Figure 2.6. When we look for instance at the third positive event (activity c) in trace pi_1, it can be seen that five artificial negative events are induced at this position. Because activity d appears at the same position of event c_p in another similar trace in the event log, namely trace pi_2, activity d cannot be added as a negative event in trace pi_1 at the position of event c_p. In contrary, all the other activity types in this event log can be added as artificial negative events because there are no similar traces in the event log where these activities appear as a positive event at the position of event c_p in trace pi_1. In this way, artificial negative events are added to the event log for each positive event in a log trace (see Table 2.6).

pi_1	abcdeg
pi_2	abdceg
pi_3	abcdefg
pi_4	abdcefg

Figure 2.5: Example event log

2.4.2 A Precision Metric Based on Artificially Generated Negative Events

The availability of an event log supplemented with artificial negative events allows for the construction of a confusion matrix for a mined control-flow model. By

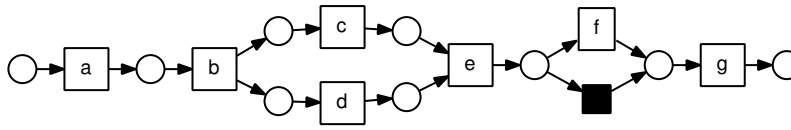


Figure 2.6: Process Model for the event log in Figure 2.5

pi_1	a_p	b_p	c_p	d_p	e_p	g_p	
	b_n	a_n	a_n	a_n	a_n	a_n	
	c_n	c_n	b_n	b_n	b_n	b_n	
	d_n	d_n	e_n	c_n	c_n	c_n	
	e_n	e_n	f_n	e_n	d_n	d_n	
	f_n	f_n	g_n	f_n	f_n	e_n	
	g_n	g_n		g_n	g_n	f_n	
pi_2	a_p	b_p	d_p	c_p	e_p	g_p	
	b_n	a_n	a_n	a_n	a_n	a_n	
	c_n	c_n	b_n	b_n	b_n	b_n	
	d_n	d_n	e_n	d_n	c_n	c_n	
	e_n	e_n	f_n	e_n	d_n	d_n	
	f_n	f_n	g_n	f_n	f_n	e_n	
	g_n	g_n		g_n	g_n	f_n	
pi_3	a_p	b_p	c_p	d_p	e_p	f_p	g_p
	b_n	a_n	a_n	a_n	a_n	a_n	a_n
	c_n	c_n	b_n	b_n	b_n	b_n	b_n
	d_n	d_n	e_n	c_n	c_n	c_n	c_n
	e_n	e_n	f_n	e_n	d_n	d_n	d_n
	f_n	f_n	g_n	f_n	f_n	e_n	e_n
	g_n	g_n		g_n	g_n	g_n	f_n
pi_4	a_p	b_p	d_p	c_p	e_p	f_p	g_p
	b_n	a_n	a_n	a_n	a_n	a_n	a_n
	c_n	c_n	b_n	b_n	b_n	b_n	b_n
	d_n	d_n	e_n	d_n	c_n	c_n	c_n
	e_n	e_n	f_n	e_n	d_n	d_n	d_n
	f_n	f_n	g_n	f_n	f_n	e_n	e_n
	g_n	g_n		g_n	g_n	g_n	f_n

Table 2.6: Negative events for the traces in the example event log

replaying the event log in the model, each and every positive and artificial negative event can be evaluated. In this way, a matrix as in Table 2.7 can be constructed denoting whether the positive and negative events are predicted correctly or not by the model. The availability of such a confusion matrix is a substantial advance of the evaluation method based on the artificial negative event generation proposed by Goedertier et al. [63].

	Actual pos.	Actual neg.
Pred. pos.	True Pos. (TP)	False Pos. (FP)
Pred. neg.	False Neg. (FN)	True Neg. (TN)

Table 2.7: Confusion matrix

Drawing upon this confusion matrix, a novel precision metric is defined. By evaluating the true positive events (TP) in respect to all predicted positive events ($TP + FP$), the precision dimension of a mined process model can be assessed. Therefore we define Behavioral Precision p_B, ranging between 0 and 1, as in Equation 2.6.

$$p_B = \frac{\sum_{i=1}^{k} n_i TP_i}{\sum_{i=1}^{k} n_i TP_i + \sum_{i=1}^{k} n_i FP_i} \tag{2.6}$$

Note that k is the total number of different grouped process instances in the event log. Index i runs over all different grouped process instances. n_i is the number of instances within one group of similar process instances. TP denotes the correctly predicted positive events, while FP denotes the incorrectly predicted artificial negative events.

This precision metric, which fully coincides with the standard definition of precision within the field of data mining, has the advantage of requiring less computational resources in respect to other precision metrics mentioned earlier (i.e. it does not require a full state space exploration). Furthermore, having now both a recall metric and a precision metric based on artificially generated negative events, we are able to define an F-measure for evaluating discovered process models, as discussed in the next section.

2.4.3 The F-score for Process Discovery Evaluation

Originally, the F-measure (Equation 3.1) was proposed by van Rijsbergen [147] in the context of information retrieval. In the fields of machine learning and data mining [123], the F-measure is often used as a standard balance between precision and recall for evaluating point classifiers.

$$F_\beta = \frac{(1+\beta)^2 \times precision \times recall}{\beta^2 \times precision + recall} \tag{2.7}$$

In fact, the F-measure can be seen as a point classifier alternative of the AUC (Area Under the ROC-curve) [51], a popular evaluation metric for rank classifiers. AUC cannot be used as an evaluation approach for process discovery because a discovered process model can only be seen as a point classifier for each individual event in the event log. In particular, a discovered process model determines whether a positive or artificial negative event is correctly classified or not, it does not assign a probability to this classification.

The β-parameter is a weight factor which determines the relative importance of precision and recall. With β equal to 1, precision and recall are weighted evenly. This results in the F1-score. However, for process discovery in real-life environments, it might be interesting to change the weight factor in favor of recall. As such, in the next chapter, we will also report on the F2-score, weighing recall twice as much as precision. This is because especially for real-life event logs, recall tends to be more important than precision.

In order to take into account the typicalities of the process discovery evaluation setting, we define the Behavioral F-measure $F1_{r_B^p, p_B}$ for a discovered process model in Equation 2.8, entirely founded upon the formula in Equation 3.1.

$$F1_{r_B^p, p_B} = 2 \times \frac{p_B \times r_B^p}{p_B + r_B^p}, \quad with \tag{2.8}$$

$$p_B = \frac{\sum_{i=1}^{k} n_i \, TP_i}{\sum_{i=1}^{k} n_i \, TP_i + \sum_{i=1}^{k} n_i FP_i}$$

$$r_B^p = \frac{\sum_{i=1}^{k} n_i \, TP_i}{\sum_{i=1}^{k} n_i \, TP_i + \sum_{i=1}^{k} n_i FN_i}$$

Note that the meaning of the symbols remain exactly the same as in Equation 2.6, with *FN* denoting the falsely predicted positive events.

The key advantage of this novel evaluation approach consists of a transparent and robust method to combine two important evaluation dimensions for discovered process models: recall and precision. More precisely, our approach allows for careful comparison of different process models obtained from the same event log. As such, benchmarking state-of-the-art process discovery techniques can be carried out in an understandable and effective way. What is more, we think that this novel evaluation approach is an important step towards a well-defined evaluation framework for process discovery.

Also, the availability of a new precision metric, quantifying whether a process model does not underfit the data, is of major importance. The Behavioral Precision (p_B) is theoretically sound and can be calculated swiftly. In this way, this metric is a useful alternative for the currently available precision metrics, that suffer from computational inefficiencies.

In order to empirically validate our proposed evaluation approach, we will demonstrate the application of the Behavioral F-measure ($F1_{r_B^p, p_B}$) within a benchmarking experiment based on 20 artificial event logs. This analysis is presented in the next section.

2.4.4 An Illustrative Example

Figure 2.7 illustrates the novel evaluation approach with a simple example. For an event log 2.7a and three corresponding control-flow models 2.7b 2.7c 2.7d, two evaluation approaches are compared. Approach A is our novel evaluation approach based on artificially generated negative events enabling the computation of $F1_{r_B^p, p_B}$. The other evaluation approach consists in applying the same F-measure to two other popular process mining metrics, namely fitness and advanced behavioral appropriateness (Rozinat et al. [107]). As can be seen from Table 2.7e, both approaches correctly evaluate the best process model and punish the imprecise flower model and the incomplete model 2.7d. However, there exist some small differences between the two approaches. This discrepancy is mainly due to the differences in how precision is quantified. p_B depends on replaying the event log in the discovered Petri net model, while $a_B^{'}$ only takes into account the ratios of sometimes follows and sometimes precedes relations in the model and in the event log.

2.5 Discussion

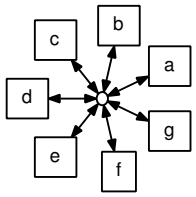
2.5.1 Completeness Assumption

As explained earlier, the proposed evaluation approach is entirely based on the principle of inducing artificial negative events into an event log. In order to induce these negative events, we make use of the assumption that an event log portrays more or less the complete behavior of the underlying process. It is acknowledged that in some real-life situations, it cannot be expected that the complete behavior of the underlying process is captured in an event log. Accordingly, this strong completeness assumption might put a strain on the applicability of artificial negative event-based evaluation metrics.

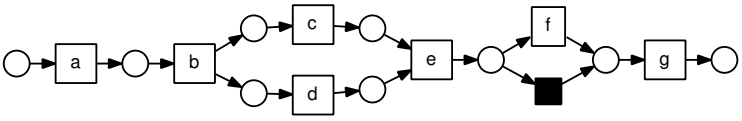
However, it is argued that the completeness assumption is an inherent problem for many unsupervised data mining tasks. When building a model based on a

pi_1	abcdeg
pi_2	abdceg
pi_3	abcdefg
pi_4	abdcefg

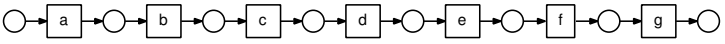
(a) Event log



(b) An overly general model



(c) An accurate and precise model



(d) An incomplete model

	Evaluation approach A			Evaluation approach B		
	r_B^p	p_B	$F1_{r_B^p, p_B}$	f	$a_B^{'}$	$F1_{f, a_B^{'}}$
Best Model	1	1	1	1	1	1
Flower Model	1	0,15	0,26	1	0,10	0,18
Incomplete Model	0,85	0,76	0,80	0,87	1	0,93

(e) Evaluation results

Figure 2.7: Illustration of the novel evaluation approach

certain data set, it is always complicated to induce models that generalize towards unseen behavior that is not represented in the data. This is also the case for process discovery and the according definition of process discovery evaluation metrics.

Secondly, completeness of an event log is strongly determined by the size of the log and by the time frame of the event log under consideration. Furthermore, the type of process is a very important determinant on whether one can assume that the behavior in an event log is to some extent complete. In many situations, only domain experts will be able to assess to what extent a certain event log is incomplete.

Nevertheless, we recognize that in highly flexible environments, it might be the case that an event log does not capture all possible behavior of a deployed process. However, we think that our novel evaluation approach is definitely of added value for the evaluation of discovered process models. Of course, when applying our approach, one should always bare in mind the consequences of the completeness assumption. When you are investigating highly unstructured processes, it might be necessary to also include other evaluation techniques. Furthermore, it is pointed out that the definition of the F-score allows to weight recall and precision differently. By varying the weight parameter β, it is possible to attach less weight to precision. By doing so, the impact of the completeness assumption is reduced because only the precision component of the F-score is influenced because of the inclusion of falsely predicted negative events which could be falsely introduced in the event log.

Finally, the ideal solution for adapting the F-score evaluation methodology to the problem of a strong completeness assumption is an improvement of the induction procedure of the artificial negative events. In [148], a first step towards improving artificial negative event generation was proposed. As such, it might become possible to alleviate the completeness issue which would make our evaluation approach even more generally applicable.

2.5.2 Root Cause Analysis

Identification of the root causes of control-flow inaccuracies of a certain discovered process model is an issue that is not addressed in this chapter. Because the proposed evaluation method is not yet implemented as a ProM plug-in, we cannot provide a means for root cause analysis. However, it should be possible to devise a plug-in based on the novel evaluation approach that visually represents flaws in the process model under investigation. For the fitness and advanced behavioral appropriateness, there exists a plug-in in the ProM framework, called Conformance Checker, that allows to some extent the identification of root causes of control-flow inaccuracies. However, in the latest version of ProM, i.e. ProM 6, no conformance checking plug-ins are available that visually indicate the root causes

of inaccurate discovered process models.

2.5.3 Overfitting: Generalization Dimension

As explained in Section 2.3, precision gauges whether a model underfits the data. However, the proposed evaluation method does not allow for the verification to what extent a process model overfits the data. Our evaluation approach is not able to detect whether a process model overfits the data by for example just enumerating all the traces in the event log. Rozinat et al. [107] defined the advanced structural appropriateness as a metric to determine whether a process model overfits the data. This metric takes into account structural properties of the mined process models in terms of alternative duplicate tasks and redundant invisible tasks. Currently, quantifying the generalization dimension based upon artificially generated negative events seems unfeasible. This causes our approach to be not fully applicable yet as a general evaluation framework for process discovery. Note that generalization metrics are subject of ongoing research. For instance in [136], the authors propose a new approach for quantifying generalization. However, this metric is yet to be implemented. In future work, we will investigate how we can quantify overfitting process models and thus further improve upon the important problem of finding a balance between generalization and precision.

Chapter 3

A Multi-Dimensional Quality Assessment of Process Discovery Techniques

Process mining techniques are specifically designed to provide insights into the real operational semantics of business processes. The distinctive, analytical focus on business processes makes process mining a research field on its own. The difference between process mining and traditional business intelligence (BI) tools is eminent. The added value of process mining over BI and other OLAP reporting tools lies in the depth of the analysis as BI tools mainly focus on the display of Key Performance Indicators (KPIs). The major drawback of classical BI tools is that they lack the ability to provide thorough insight into the root causes of process inefficiency and erroneousness.

Process discovery is one such technique which provides a process analyst with this profound level of insight into the actual way of working. The graphical visualization offered by process discovery techniques is definitely one of the major driving forces of the success of process mining in practice. However, knowledge discovery in real-life event logs faces the crucial requirement to extract useful, interpretable information, preferably in line with expectations of domain specialists. Research with respect to the application of process discovery techniques in real-life settings has indicated that the currently available techniques often suffer from incomprehensibility.

In this chapter, the quality of existing process discovery techniques is thoroughly assessed. As such, the major contributions are:

- a comprehensive overview of process discovery techniques
- the first comparative benchmarking study of process discovery techniques using multiple real-life event logs
- a multi-dimensional evaluation study focussing on both accuracy and comprehensibility
- demonstration of the F-score as an evaluation approach for combining accuracy dimensions

The chapter is organized as follows. The first section provides a comprehensive literature review of process discovery techniques. Section 3.3 presents a discussion on how to evaluate process discovery techniques with respect to accuracy. Furthermore, in Section 3.4, we outline a number of process model complexity metrics in order to assess the comprehensibility of discovered process models. Then, Section 3.5 presents the empirical setup of the benchmarking experiment before the results are discussed in Section 3.6. The last sections are devoted to a discussion on the aspect of representational bias and conclusions.

3.1 Discussion of Process Discovery Techniques

In the past decade and a half, many process discovery techniques have been proposed (see Table 3.1). Algorithmic, machine learning as well as probabilistic approaches have been conceived. One of the major drivers for process miners is the development of the ProM framework [141]. This supporting framework developed at TU/Eindhoven allows the development of all kind of process intelligence techniques. One of its most important features is the underlying file format for process event logs, formerly known as MXML, but recently improved to XES (eXtensible Event Stream).

3.1.1 Early Approaches

The foundational approaches to process discovery were formulated by Agrawal [5], Cook and Wolf [33], and Datta [35]. Cook and Wolf proposed three different approaches in the context of software engineering. Their RNet, Ktail and Markov methods approached the discovery of models from event-based data from a statistical, algorithmic and probabilistic viewpoint. Nonetheless, the idea of applying process discovery in the field of workflow management systems stems from Agrawal et al. and Datta.

Manilla and Meek [88] describe a technique to learn two-component mixture models of global partial orders that provide an understandable, global view of a set

of sequences. As a result, the technique cannot cope with concurrency and many other typical problems in the field of process mining. Another process mining tool was developed by Schimm [114, 115]. His Process Miner is entirely based on data mining techniques and is able to discover complete and minimal process schemes from event-based data. Yet, Process Miner cannot be qualified as robust, which is an important requirement in order to apply techniques in real-life settings.

3.1.2 The α-algorithm and Its Successors

The α-algorithm can be considered as one of the most substantial techniques in the process mining field. Van der Aalst et al. [131] prove that it can learn an important class of workflow nets, called structured workflow nets, from complete event logs. The α-algorithm assumes event logs to be complete with respect to all allowable binary sequences and assumes that the event log does not contain any noise. Therefore, the α-algorithm is sensitive to noise and incompleteness of event logs. Moreover, the original α-algorithm was incapable of discovering short loops or non-local, non-free choice constructs. Alves de Medeiros et al. [9] improved the original α-algorithm to mine short loops and named it α^+. This algorithm is available in ProM and therefore, we included this algorithm in Table 3.1.

More techniques have been proposed to improve the original α-algorithm. Wen et al. [154] devised the α^{++}-algorithm that is able to detect non-free choice constructs. Furthermore, Wen et al. [155] have also proposed to take advantage of both start and completed event types in order to detect concurrency, which is implemented in their β-algorithm.

In order to remedy the robustness problem of the α-algorithms, Weijters et al. [153] developed HeuristicsMiner. In particular, HeuristicsMiner extends the formal α-algorithm in that it applies frequency information with regard to three types of relationships between activities in an event log: direct dependency, concurrency and not-directly-connectedness. According to its specification, HeuristicsMiner can discover short loops and non-local dependencies, but lacks the capability of detecting duplicate activities. As a result of the threshold parameter setting, the heuristic algorithm is prone to be noise resilient and therefore can be expected to be robust in a real-life context.

In contrast to the theoretically grounded α-algorithm, Günther and van der Aalst [67] propose Fuzzy Miner, an adaptive simplification and visualization technique based on significance and correlation measures to visualize the behavior in event logs at various levels of abstraction. The main contribution of Fuzzy Miner is that it can also be applied to less structured, or unstructured processes of which the event logs cannot easily be summarized in concise, structured process models. Although this approach is an extraordinary data exploration technique, it suffers from a drawback in the sense that a Fuzzy model cannot be translated to a

Name	Author(s)	Year	Typical problems					Implemented in ProM	Approach						Assessed in this study
			Noise	Duplicate tasks	Hidden tasks	Non-free choice constructs	Loops		Genetic algorithm	Classification-based	Clustering-based	Algorithmic approach	Inductive Logic Progr.	Probabilistic approach	
General DAG	Agrawal et al. [5]	1998	✓		✓							✓			
B-F(k,c)-algorithm	Datta [35]	1998	✓				✓					✓			
Rnet, Ktail, Markov	Cook & Wolf [33]	1998	✓		✓		✓			✓		✓		✓	
Global partial orders	Manilla & Meek [88]	2000	✓					✓				✓		✓	
Process Miner	Schimm [114, 115]	2002		✓			✓			✓	✓				
α/α^+	van der Aalst et al. [9, 131]	2004						✓				✓			✓
InWoLvE - splitpar	Herbst & Karagiannis [71]	2004	✓	✓	✓							✓		✓	
Multi-phase Miner	van Dongen & van der Aalst [146]	2005						✓				✓			
WorkflowMiner	Gaaloul et al. [60]	2005						✓				✓			
Heuristics Miner	Weijters et al. [152, 153]	2006	✓		✓	✓	✓	✓				✓			✓
DWS Mining	Greco et al. [66]	2006	✓			✓		✓		✓	✓				
Rule-based approach	Maruster et al. [91]	2006	✓							✓					
-	Ferreira & Ferreira [56]	2006											✓		
Genetic Miner	Alves de Medeiros et al. [10]	2007	✓		✓	✓	✓	✓	✓						✓
DT Genetic Miner	Alves de Medeiros et al. [7]	2007	✓	✓	✓	✓	✓	✓	✓						✓
Fuzzy Miner	Günther & van der Aalst [67]	2007	✓				✓	✓			✓	✓			
α^{++}	Wen et al. [154]	2007				✓		✓				✓			✓
DecMiner	Lamma et al. [81]	2007						✓					✓		
AWS Mining	Greco et al. [65]	2008	✓				✓	✓			✓				
AGNEsMiner	Goedertier et al. [63]	2009	✓	✓	✓	✓	✓	✓		✓			✓		✓
β (or Tshingua α)	Wen et al. [155]	2009					✓	✓				✓			
Enhanced WFMiner	Folino et al. [57]	2009	✓	✓	✓	✓	✓	✓				✓			
EM-approach	Ferreira & Gilblad [53]	2009	✓											✓	
ILP Miner (Parikh)	Van der Werf et al. [145]	2009			✓			✓				✓			✓
FSM Miner/Petrify	van der Aalst et al. [128, 139]	2010		✓	✓	✓	✓	✓				✓			
FSM Miner/Genet	Carmona et al. [27]	2010					✓					✓			
SMT C-net discovery	Solé & Carmona [119]	2012					✓					✓			

Table 3.1: Overview of process discovery techniques

formal Petri net which severely limits a comparative evaluation to other process discovery techniques.

3.1.3 Techniques Originating from Machine Learning Theory

Many authors have proposed to use machine learning techniques in the context of control-flow discovery. Several authors have used classification techniques for the purpose of process discovery. For instance, Maruster et al. [91] were among the first to investigate the use of rule-induction to predict dependency relationships between activities from a corpus of reference logs that portray various levels of noise and imbalance. To this end, the authors use a propositional rule induction technique, i.e. the uni-relational classification learner RIPPER, on a table of direct metrics for each process task in relation to each other process task, which is generated in a pre-processing step.

Ferreira and Ferreira [56] apply a combination of ILP learning and partial-order planning techniques to process mining. By iteratively combining planning and learning, a process model is discovered that is represented in terms of the case data preconditions and effects of its activities. In addition to this novel process mining technique, the contribution of this work is in the truly integrated BPM life cycle of process generation, execution, re-planning and learning. Lamma et al. [81] also describe the use of ILP to process mining. The authors assume the presence of negative sequences to guide the search algorithm. Unlike the approach of Ferreira and Ferreira, who use partial-order planning to present the user with an execution plan to accept or reject (a negative example), this approach does not provide an immediate answer to the origin of the negative events.

Due to the limitations of local search, early approaches to process discovery, were generally not able to discover non-trivial constructs like non-free choice, invisible tasks and duplicate tasks. The main motivation of Alves de Medeiros et al. [10] to apply a genetic algorithm for process discovery is to benefit from global search. The fitness function of this genetic algorithm incorporates both a recall and a precision measure that drives the genetic algorithm towards suitable models. Genetic Miner defines its search space in terms of causal matrices. These matrices express task dependencies only, yet they are closely related to Petri nets. Because of the global search property, Genetic Miner is capable of detecting non-local patterns in the event log. Moreover, the algorithm ensures a fair robustness because of the arc post-pruning step.

AGNEsMiner [63], proposed by Goedertier et al., addresses the difficulties of process mining by representing the discovery task as first-order classification learning on event logs, supplemented with artificially generated negative events. Like Genetic Miner, the AGNEs algorithm is capable of constructing Petri net mod-

els from event logs and has been implemented as a mining plugin in the ProM Framework. Given an event log supplemented with artificially generated negative events, it becomes possible to learn the conditions that discriminate between the occurrence of either a positive or a negative event. As such, process mining is represented as a classification learning problem. The AGNEs process discovery algorithm is designed to learn the discriminating conditions that determine whether an event can take place or not, given a history of events of other activities. To induce this knowledge, AGNEsMiner makes use of an existing multi-relational classification learner called Tilde [13].

Another approach based on machine learning theory was proposed by Greco et al. [66]. This technique, called DWS mining, can be described as a hierarchical and iterative procedure that refines the process model in each step, based on clustering of patterns sharing similar behavior. This approach guarantees full compliance with the event log and increasingly improves the soundness of the process model, where soundness can be seen as a quantification of precision. In [65], Greco et al. extend traditional discovery methods by putting forward an abstraction method that aims to produce a taxonomy of process models. As such, the behavior in the event log can be analyzed at different levels of detail. AWS mining consists of both mining and abstraction algorithms that are assembled in order to build a tree-like schema. This schema consists of non-leaf nodes with an abstract process model that generalizes all the different process models in the corresponding subtree.

The idea to represent mined process models through different views at different levels of abstraction is also the driver behind FSM Miner/Petrify, a process discovery technique proposed by van der Aalst et al. [128, 139]. In search of finding a balanced trade-off between precise and general process models, the authors propose a two-step approach. Firstly, a transition system should be constructed from the traces in an event log. In a second step, this transition system is synthesized by means of theory of regions. In this way, a Petri net can be constructed. Due to the more direct relation between the log and the transition system, generalization is more controllable. An important disadvantage of this approach is that it cannot deal with noise. A similar technique is described by Carmona et al. [27]. This algorithm, called Genet, also enables the construction of a Petri net from a transition system.

3.1.4 Other Process Discovery Approaches

The last part of this section enumerates some other process discovery techniques that were not described previously.

Herbst and Karagiannis [71] describe the working of the splitpar algorithm that is part of the InWoLvE framework for process analysis. This algorithm derives a so-called stochastic activity graph and converts it into a structured process

model. The splitpar algorithm is capable of detecting duplicate activities, but it is not able to discover non-local dependencies. Furthermore, Gaaloul et al. [60] propose a process discovery algorithm they called Workflow Miner. In fact, this technique enables the discovery of advanced structural workflow patterns after modeling the elementary dependencies in the event log in an intermediary graphical representation.

Furthermore, van Dongen and van der Aalst showed in [146] how process discovery can be approached by aggregating individual process instance models into a Petri net. For this approach, they rely on the use of Event-driven Process Chains (EPCs) as an intermediate step in order to derive an executable process model.

Another approach to process mining is the Enhanced WFMiner (Folino et al. [57]). This algorithmic technique is described as being capable of dealing with a large number of important process discovery challenges such as noise, duplicate tasks and non-free choice. Moreover, Ferreira and Gillblad [53] address the problem of unlabeled event logs, which are logs without a case identifier for each process execution. By applying an Expectation-Maximization procedure, their technique allows to discover process models from an unlabeled stream of events.

Process discovery can also be formulated as a mathematical optimization problem. For instance, ILP Miner [145] entails the application of Integer Linear Programming (ILP) to automatically discover Petri nets. This technique, formerly known as Parikh language-based region miner, is based on the well-known theory of regions and since this approach allows for parallelization and is shown to be independent of the number of events registered in the event log, the technique can be expected to be useful in practice. In a similar way, Solé and Carmona [119] recently proposed the use of Satisfiability Modulo Theories (SMT) for process discovery. In contrast to [145], these authors do not make use of ILP but instead solve the problem in the boolean domain by means of an SMT-solver. Note that the latter technique discovers C-nets instead of Petri nets.

3.2 Evaluation Framework

The assessment of process discovery techniques centers on accuracy and comprehensibility, as depicted in Figure 3.1. The former refers to how well a process discovery technique is able to render process models that capture the behavior in an event log soundly, hereby balancing between overly general or overly precise models. In contrast, comprehensibility entails the assessment of the understandability of the discovered process models in terms of their complexity and ease of interpretation.

In the following two sections, it is outlined how both general evaluation dimen-

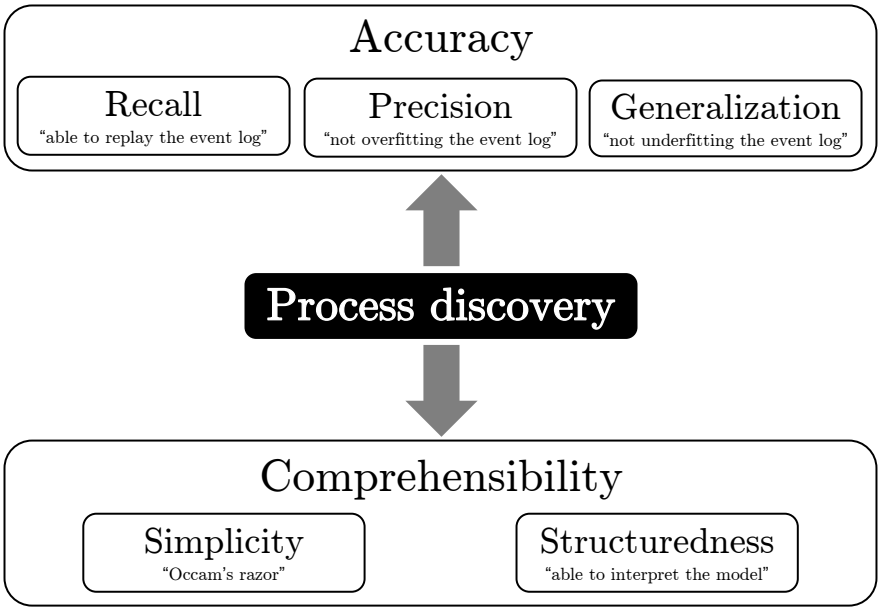


Figure 3.1: Overview of the quality dimensions of a discovered process model

sions are quantified in order to assess the quality of process discovery techniques. It should be noted that with respect to accuracy, no performing generalization metric is currently available for process discovery evaluation, especially in case of real-life event logs. Accordingly, accuracy evaluation will only take into account recall and precision perspectives.

3.3 Evaluating Accuracy

Since the purpose of this chapter is a multi-dimensional quality assessment of process discovery techniques, appropriate metrics need to be at hand in order to conduct this evaluation. In the previous chapter, an overview of how discovered process models can be judged with respect to their accuracy was presented. As mentioned earlier, accuracy in itself is multi-dimensional, which brings about a multitude of available metrics. Furthermore, in order to compare different process discovery techniques, it is indispensable to have some kind of methodology to combine accuracy evaluation dimensions. In this section, it is argued that the F-score [147] can be employed in order to combine different types of accuracy metrics.

3.3.1 Accuracy Dimensions

The accuracy of a process model is a somewhat abstruse concept because it can only be assessed taking into account different perspectives. As such, we identify three dimensions that are important: recall, precision, and generalization. It should be noted that process discovery techniques should be able to meet multiple criteria at the same time. For instance, techniques that only strive for maximizing recall, tend to provide either overfitting models that overspecify the data in the event log or underfitting models that allow much more behavior than actually desired. Therefore, the key challenge for any process discovery technique is to discern process models that are able to find a balance between multiple dimensions in order to provide useful models.

As described in the previous chapter, a variety of model-log metrics are available for quantifying discovered process model accuracy. Table 3.2 provides an overview of the metrics that are currently available for the accuracy assessment of process discovery techniques. In the empirical assessment, multiple of these metrics will be employed so as to counterbalance the problem that many process discovery evaluation metrics suffer from different drawbacks. Together with the large number of event logs used for the comparative assessment, this should stimulate the objectivity of the results.

Dimension	Name	Symbol	Author	Available in ProM	Range	Model input type
Recall	Fitness	f	Rozinat and van der Aalst [107]	✓	[0,1]	Petri net
	Completeness	$PF_{complete}$	Alves de Medeiros et al. [7]	✓	[-∞,1]	Heuristic net
	Completeness		Greco et al. [66]		[0,1]	Workflow schema
	Parsing Measure	PM	Weijters et al. [153]	✓	[0,1]	Heuristic net
	Succesfully executed traces	set	Rozinat [109]	✓	[0,1]	Petri net
	Behavioral Recall	r_B^p	Goedertier et al. [63]		[0,1]	Petri net
	Average alignment-based trace fitness	f_a^{avg}	Adriansyah et al. [4]	✓	[0,1]	Petri net
Precision	Behavioral Appropriateness	a_B	Rozinat and van der Aalst [107]	✓	[0,1]	Petri net
	Advanced Behavioral Appropriateness	$a_B^{'}$	Rozinat and van der Aalst [107]	✓	[0,1]	Petri net
	Soundness		Greco et al. [66]		[0,1]	Workflow schema
	Behavioral Specificity	s_B^n	Goedertier et al. [63]		[0,1]	Petri net
	Behavioral Precision	p_B	De Weerdt et al. [43]		[0,1]	Petri net
	ETC Precision	etc_P	Muñoz-Gama and Carmona [96]	✓	[0,1]	Petri net
	One align precision	a_p^1	Adriansyah et al. [1]	✓	[0,1]	Petri net
	Best align precision	a_p	Adriansyah et al. [1]	✓	[0,1]	Petri net

Table 3.2: Overview of process discovery accuracy dimensions and metrics

Combining Precision and Recall: F-score

A crucial problem of a comparative quality assessment of process discovery techniques is the absence of a validated evaluation framework for discovered process models. In this study, the F-score is employed in order to evaluate the overall accuracy of a discovered process model. It should be noted that there is no proper generalization metric available. However, the availability of multiple precision metrics allows to investigate the way process models underfit the data. Since underfitting

is typically a more common problem for process discovery techniques, the proposed methodology remains a defensible approach. Nevertheless, it is pointed out that in an ideal scenario, generalization is also taken into account. In this way, a more comprehensive evaluation methodology could be set up.

Originally, the F-measure (Equation 3.1) was proposed by van Rijsbergen [147] in the context of information retrieval. In the fields of machine learning and data mining [123], the F-measure is often used as a standard balance between precision and recall for evaluating point classifiers.

$$F_\beta = \frac{(1+\beta)^2 \times \text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}} \tag{3.1}$$

The β-parameter is a weight factor to influence the relative importance of precision and recall. With β equal to 1, precision and recall are weighted evenly. This results in the F1-score. However, because process models discovered from real-life event logs are used, it might be interesting to change the weight factor in favor of recall. In Section 4.5, we will also report on the F2-score, weighing recall twice as much as precision. This is because especially for real-life event logs, recall is to be considered more important than precision.

3.4 Evaluating Comprehensibility

As an evaluation criterion for mined process models, comprehensibility has received modest attention in the literature. Nonetheless, some articles have indicated that the available process discovery algorithms face a crucial problem with discovering comprehensible process models from highly unstructured event logs [62, 68, 139]. An important reason for the inferior interest in comprehensibility is the lack of good quantification methods for process model complexity. Moreover, the issue of comprehensibility is of inferior value when taking into account artificial event logs. Typically, process models mined from such event logs are smaller and much less complex then models mined from real-life event logs.

Within the domain of process modeling, Mendling et al. [93] have done much empirical work on the understandability of process models. They characterize understandability in terms of personal, structural and textual factors. They also found that experts with a more elevated familiarity with concurrency typically are better in understanding process models. Furthermore, an important insight of their study is the relation between structuredness and understandability. The most important result for our study is the major influence of model size on understandability.

3.4.1 Comprehensibility Metrics

Although Mendling et al. describe interesting results with respect to the comprehensibility of process models, they do not provide one or more comprehensive metrics that quantify understandability. Therefore, we will employ metrics that were proposed by Lassen and van der Aalst. In [82], they describe three metrics for quantifying model complexity: Extended Cardoso metric (ECaM), Extended Cyclomatic metric (ECyM) and Structuredness Metric (SM). The first metric is based on a metric proposed by Cardoso [26] which takes into account certain splits and joins within the process model. The ECaM metric inherently perceives process model complexity in a local way because the metric only takes into account the successor nodes of each place. The ECyM metric builds further on work of McCabe [92]. In contrast to ECaM, the ECyM metric is based on the state space of the process model. This characteristic puts a strain on its applicability in a real-life environment because calculating the state space of complex process models often turns out to be infeasible. Finally, SM is a new metric that identifies behavioral patterns in a process model and scores these patterns according to their type and elements. The authors state that their SM metric is a better reflection of complexity because it takes into account penalties for components that are embedded in others. Lassen and van der Aalst implemented their metrics in a plugin for the ProM-framework. It should be pointed out that these metrics are designed for safe and sound WF-nets. The main reason is that the metrics are developed in the context of process modeling, rather than process mining. Because most process mining algorithms cannot guarantee to produce safe and sound WF-nets, we should take this limitation into account in the results section.

Next to ECaM, ECyM and SM, we will assess process model comprehensibility by also reporting more straightforward indications of model complexity in terms of Petri net building blocks such as number of transitions, number of places and number of arcs. Furthermore, the amount of a number of control-flow constructs are reported: the number of AND-Splits/Joins and the number of OR-Splits/Joins. Note that the inclusion of these measurements is driven by the findings of Mendling et al. [93] who showed that model size has a major impact on understandability. It should be noted that these net characteristics are influenced by the type of algorithm. For example, the genetic algorithms and HeuristicsMiner produce Heuristic nets. When these nets are converted into Petri nets, a lot of invisible transitions are created, mainly for routing purposes. This creates a certain bias with respect to employing these metrics for comprehensibility evaluation.

3.5 Empirical Setup

The first and major objective of this study is to assess process discovery techniques with respect to accuracy and comprehensibility. In order to do so, we will apply most of the metrics presented in Sections 3.3 and 3.4 to a subset of techniques depicted in Section 3.1. Furthermore, we will apply advanced statistical analysis techniques in order to compare the results and draw general conclusions. The empirical setup of the study is described in the following paragraphs.

3.5.1 Process Discovery Techniques

We assessed a total of seven state-of-the-art process discovery techniques: α^+, α^{++}, AGNEsMiner, Genetic Miner, Duplicate Task (DT) Genetic Miner, HeuristicsMiner and ILP Miner. Furthermore, we also report on the results of a flower model for each of the event logs, which is a process model that allows for any combination of activities and thus is a maximum sensitive but highly imprecise process model. It was infeasible to incorporate all process discovery techniques from Section 3.1 because of three important reasons. First of all, not every proposed algorithm is publicly available. Secondly, in order to evaluate process models, it is required that the technique presents results in the form of a Petri net or that the result is transformable to a Petri net. Finally, many techniques lack scalability and therefore don't allow to calculate models based on real-life event logs which are typically larger and more complex than artificially constructed event logs.

Suitability of the Search Space: Representational Bias

It is recognized that the current experimental setup considers Petri nets as being the representational language for the discovered process models. This choice is mainly driven by the fact that a majority of process discovery techniques makes use of this representation or their representation can be translated into a Petri net. A second element that plays an important role in this choice relies in the dominance of Petri nets as the underlying model representation for conformance checking techniques.

However, as indicated in [133], Petri nets suffer from inherent drawbacks as a representational language for process discovery. For instance, the search space includes large numbers of inconsistent models (e.g. presenting deadlocks) which is an undesirable feature of the representational bias. Furthermore, because of the low-level nature of Petri nets, each specific algorithm makes further assumptions with regard to the representational bias. As an example, some algorithms support invisible transitions, others support duplicates, loops, etc. Accordingly, it is often the process itself in terms of its characteristics that determines which representational bias is most appropriate [134]. However, in real-life environments, it

is a distinct challenge to upfront determine which representational bias is more suitable. Thus far, developers of process discovery algorithms did not explicitly consider the assumptions made with regard to the representational bias. As proposed in [133], a more tailored representational language for process discovery such as C-nets are capable of alleviating the bias problem. Nevertheless, the adoption of C-nets in the process mining domain is still limited.

Parameter Settings

Although many of the selected techniques require parameter tuning, we opted to employ standard settings as much as possible. However, for Genetic Miner and DT Genetic Miner, we reduced the population size to 10 and increased the maximum number of generations to 5.000 for the real-life event logs. This is according to the experiments in [10]. Furthermore, HeuristicsMiner was configured to also discover long distance dependencies. Finally, for AGNEsMiner, the injection probability π was lowered to 0,04 consistent with [62].

It should be noted that parameter tuning of some of these algorithms could potentially increase the performance. However, changing the parameters often requires prior knowledge that is seldomly available. Due to relatively high calculation times for both the process models as well as the evaluation metrics, especially for the more complex real-life event logs, a parameter tuning phase is practically infeasible. Moreover, for techniques like Genetic Miner and AGNEsMiner, there are over 10 parameters. This makes the inclusion of parameter tuning even more infeasible.

3.5.2 Artificial and Real-Life Event Logs

Evaluation of process discovery algorithms traditionally relied on artificial event logs. For example, Alves de Medeiros et al. [10] and Goedertier et al. [63] report performance of different algorithms on artificial logs in order to demonstrate some specific requirements for process discovery algorithms such as dealing with parallel behavior, loops, non-local dependencies, etc. Alves de Medeiros et al. also present some results on four small scale real-life event logs. However, to our knowledge, this study is the first to provide a comprehensive benchmarking study of process discovery algorithms using real-life event logs.

In order to present a thorough assessment of process mining techniques, we will evaluate the algorithms both on artificial as well as on real-life event logs. Table 3.3 provides some details on twenty artificial event logs, which have been used previously by Alves de Medeiros et al. [7] to evaluate the Genetic Miner algorithm.

The real-life event logs are described in Table 4.1. These logs originate from information systems of different organizations: a university, a telecom company, a

	activity types	≠ process inst.	process inst.	= activity types	loop	skip	non-free choice	duplicate tasks
a10skip	12	6	300	1		✓		
a12	14	5	300	2				
a5	7	13	300	1	✓			
a6nfc	8	3	300	1			✓	
a7	9	14	300	4				
a8	10	4	300	1				
betaSimplified	13	4	300	0		✓	✓	✓
choice	12	16	300	0				
DriversLicense	9	2	300	0				
DriversLincensel	11	87	350	1	✓	✓	✓	✓
herbstFig3p4	12	32	300	3	✓			
herbstFig5p19	8	6	300	1				✓
herbstFig6p18	7	153	300	0	✓			
herbstFig6p31	9	4	300	0				✓
herbstFig6p36	12	2	300	0			✓	
herbstFig6p38	7	5	300	3				✓
herbstFig6p41	16	12	300	4				
l2l	6	10	300	0	✓			
l2lOptional	6	9	300	0	✓			
l2lSkip	6	8	300	0	✓			

Table 3.3: Artificial event log properties

manufacturing company and an insurance company. All the event logs are recordings of human-centric processes and therefore exhibit a medium to high level of unstructuredness. In order to characterize the data sets, a number of event log properties are presented in Table 4.1, such as the number of process instances (# PI), number of events (# EV), number of activity types (# AT) and the number of distinct process instances (# DPI). Furthermore, we also show three important metrics proposed by Günther [67]: level of detail (LoD), structure (ST) and mean affinity (MA). Level of detail is defined as the mean number of event classes per trace in the event log. Structure denotes the amount of observed behavior compared to the amount of theoretically possible behavior. A low value on structure indicates a more demanding challenge for process discovery techniques because it is very difficult to represent unstructured behavior in a sensitive and comprehensible process model. Mean affinity is somewhat similar to structure as this metric quantifies the mean relative overlap of direct following relations between each two traces in the event log. Again, low mean affinity values indicate an elevated diffi-

culty for process discovery techniques. From Table 4.1, it can be seen that event logs UFM and XNB are especially problematic with respect to the diversity of behavior recorded.

3.5.3 Statistical Testing

For analysis purposes, we make use of a collection of statistical techniques. These techniques provide mathematical ground for the conclusions.

ANOVA, Friedman and Bonferonni-Dunn

In order to compare the performance of multiple algorithms on manifold data sets, we make use of simple ANOVA (ANalysis Of VAriance) and a procedure described in Demšar [46]. In a first step of this procedure, the Friedman test [59] is performed which is a non-parametric equivalent of the well known ANOVA test. The null hypothesis of the Friedman test states that all techniques perform equivalent. The test statistic is defined as:

$$\chi_F^2 = \frac{12P}{k(k+1)} \left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4} \right]$$

with R_j the average rank of algorithm $j = 1, 2 \ldots k$ over P data sets. Under the null hypothesis, the Friedman test statistic is distributed according to χ_F^2 with $k-1$ degrees of freedom, at least when P and k are big enough ($P > 10$ and $k > 5$), but in case of the real-life event logs, the exact critical values are used based on an adjusted Fisher z-distribution.

If the null hypothesis of equivalent performing techniques is rejected by the Friedman test, a post-hoc Bonferroni-Dunn test [49] is applied to compare the process discovery techniques. The post-hoc Bonferroni-Dunn test is a non-parametric alternative to the Tukey test and is defined as:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6P}}$$

with critical value q_α based on the Studentized range statistic divided by $\sqrt{2}$, and an additional Bonferroni correction by dividing the confidence level α by the number of comparisons made, $\frac{\alpha}{(k-1)}$, to control for family wise testing. This results in a lower confidence level and thus in higher power. The difference in performance between the best performing technique and other techniques is significant if the corresponding average ranks differ by at least the Critical Distance (CD).

Label	Event log properties							Organization	Process description
	# PI	# EV	# AT	# DPI	LoD	ST	MA		
UFM	17.812	83.286	42	1.908	4, 11	0, 70	0, 06	Telecom company	Second-line customer-initiated processes regarding the resolution of internet and telephony service problems.
P2P	10.487	97.873	23	76	9, 29	0, 90	0, 86	KU Leuven	SAP-based workflow process for handling invoices.
KHD	1.541	8.657	18	251	5, 38	0, 74	0, 24	KU Leuven	Status flow log of the ICT Service Helpdesk process.
SIM	956	11.218	22	212	10, 41	0, 76	0, 56	Manufacturing company	International Service Helpdesk processes regarding ICT-related problems.
XBM	6.407	38.380	18	155	5, 93	0, 69	0, 36	Insurance company	Interactive information sharing and resolution solving processes with remote brokers.
XOA	2.004	12.432	49	71	6, 00	0, 95	0, 11	Insurance company	Internal process of handling calls of tender.
XNB	8.384	51.752	163	277	6, 09	0, 98	0, 04	Insurance company	Internal back-office process of administering new commercial contracts with both private as well as business clients.
XAO	614	3.631	14	36	5, 90	0, 79	0, 38	Insurance company	Back-office handling of claims regarding occupational injuries.

Table 3.4: Description of the real-life event logs with following characteristics: the number of process instances (# PI), number of events (# EV), the number of activity types (# AT), the number of distinct process instances (# DPI), level of detail (LoD), structure (ST) and mean affinity (MA)

Principal Component Analysis and Canonical Correlation Analysis

In order to provide broader insights into the multi-dimensional quality assessment, the results of both a Principal Component Analysis (PCA) and a Canonical Correlation Analysis will be reported. PCA [77] is a statistical technique that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of uncorrelated variables called principal components. Because it is one of the most popular data exploration techniques, it fits very well within the research domain of process mining, which is exploratory in nature as well.

Next to PCA, we will also employ Canonical Correlation Analysis. This data analysis technique, introduced by Hotelling [73], allows to identify commonalities between two sets of variables. Given the different evaluation dimensions and the various metrics, we will be able to provide insight into the relations between these dimensions.

3.6 Results

The results section is subdivided into three parts. First, the accuracy of process discovery algorithms on artificial event logs is discussed. In a second part, we report on the assessment of the same techniques to real-life event logs with a key focus on accuracy and comprehensibility. Finally, a statistical multivariate analysis is presented in order to come up with more general findings.

3.6.1 Artificial Event Logs

The accuracy results for the artificial event logs are summarized in Tables 3.5 and 3.6. Table 3.5 represents the average results of eight algorithms on twenty artificial event logs for eight different metrics. Note that, in this and all following tables, the subsequent notation is used: the **best** average performance is underlined and denoted in bold face for each metric. A one-tailed paired t-test was used to test the significance of the performance differences in terms of absolute results. A non-parametric procedure combining the Friedman and Bonferonni-Dunn tests was used to test the significance of performance differences in terms of the average ranks, which are denoted between brackets. Performances that are **not significantly different at the 95% level** from the top-ranking performance are tabulated in bold face. Statistically *significant underperformances at the 99% level* are emphasized in italics. Performances significantly different at the 95% level but not at the 99% level are reported in normal font.

From Table 3.5, it can be concluded that overall, there is little difference between the selected process discovery algorithms. However, a remarkable result

	Recall			Precision			
	f	r_B^p	PM	$a_B^{'}$	s_B^n	p_B	etc_P
AGNEsMiner	**0,995**	**0,998**	**0,926**	**0,813**	**0,980**	**0,924**	**0,913**
α^+	*0,969*	0,952	**0,753**	**0,873**	**0,964**	0,862	**0,918**
α^{++}	0,984	0,972	**0,823**	**0,879**	**0,982**	**0,921**	**0,952**
DT Genetic Miner	0,996	**0,999**	**0,911**	**0,778**	**0,984**	**0,914**	**0,952**
Genetic Miner	**0,998**	0,984	**0,927**	*0,737*	**0,987**	**0,936**	**0,943**
HeuristicsMiner	*0,973*	*0,959*	**0,778**	**0,809**	0,982	**0,907**	**0,945**
ILP Miner	**1,000**	**0,991**	na	**0,786**	0,972	0,873	**0,902**
Flower	1,000	1,000	1,000	0,219	0,000	0,117	0,134

Table 3.5: Average accuracy results for the artificial event logs

is that HeuristicsMiner presents a significant underperformance with respect to recall. Also, the different precision metrics present somewhat contradictory results. In Section , we have indicated some issues with $a_B^{'}$ which are due to the state space exploration that is required to calculate this metric. This study again indicates significant differences between $a_B^{'}$ and other precision metrics.

Table 3.6 presents the application of the F-measure to r_B^p, p_B (approach A) and f, $a_B^{'}$ (approach B) respectively. By examining both evaluation approaches, we can conclude that AGNEsMiner, DT Genetic Miner and Genetic Miner perform outstandingly, whereas the results of HeuristicsMiner, ILP Miner and the α-algorithms tend to indicate slight accuracy problems, given the setting of evaluation based on artificial event logs. HeuristicsMiner and the α-algorithms show an underperformance in recall, while ILP Miner accounts for less precise process models.

3.6.2 Real-Life Event Logs

In contrast to the artificial event logs, the algorithms are assessed along two dimensions, namely accuracy and comprehensibility.

Accuracy

The accuracy results for the real-life event logs are displayed in Tables 3.7 and 3.8. The former presents the aggregate accuracy results. Note that in this table, the Parsing Measure (PM) is replaced by the number of successfully executed traces (abbreviated as *set*). This metric is a Petri net alternative for the Parsing Measure, which could only be calculated for about 50% of the real-life event log

	Approach A				Approach B			
	r_B^p	p_B	$F1_{r_B^p, p_B}$	$F2_{r_B^p, p_B}$	f	$a_B^{'}$	$F1_{f, a_B^{'}}$	$F2_{f, a_B^{'}}$
AGNEsMiner	**0,998** (**3,6**)	**0,924** (**3,5**)	**0,953** (**3,4**)	**0,976** (**3,3**)	**0,995** (**3,9**)	**0,813** (**3,6**)	**0,873** (**3,6**)	**0,932** (**3,6**)
α^+	0,952 (**4,6**)	0,862 (**4,7**)	0,892 (**5,0**)	0,922 (**5,0**)	*0,969* (**4,7**)	**0,873** (**3,6**)	**0,904** (**3,8**)	**0,935** (**4,1**)
α^{++}	0,972 (**4,2**)	**0,921** (**3,7**)	**0,941** (**3,8**)	**0,957** (**3,8**)	0,984 (**4,1**)	**0,879** (**3,3**)	**0,897** (**3,6**)	**0,916** (**3,5**)
DT Genetic Miner	**0,999** (**3,6**)	**0,914** (**3,8**)	**0,949** (**3,8**)	**0,977** (**3,5**)	0,996 (**3,8**)	**0,778** (**4,1**)	**0,851** (**4,1**)	**0,922** (**4,0**)
Genetic Miner	0,984 (**3,8**)	**0,936** (**3,6**)	**0,954** (**3,6**)	**0,970** (**3,7**)	**0,998** (**3,8**)	*0,737* (**5,0**)	0,829 (**4,8**)	**0,915** (**4,6**)
HeuristicsMiner	*0,959* (**4,5**)	**0,907** (**4,3**)	0,927 (**4,3**)	**0,944** (**4,4**)	*0,973* (**4,5**)	**0,809** (**4,1**)	**0,864** (**4,0**)	**0,918** (**4,1**)
ILP Miner	**0,991** (**3,7**)	0,873 (**4,5**)	0,919 (**4,2**)	**0,958** (**4,4**)	**1,000** (**3,3**)	**0,786** (**4,4**)	**0,850** (**4,2**)	**0,916** (**4,1**)
Flower	1,000	0,117	0,208	0,392	1,000	0,219	0,349	0,556

Table 3.6: Average F1- and F2-scores for the artificial event logs, with average ranks between brackets

based process models because the calculation of the metric requires a translation from a Petri net to a heuristic net which often failed. Furthermore, the etc_P-metric is not represented for the same reason since it could not be calculated for a large enough number of process models. Finally, we also added the average run time of each algorithm. Note also that the detailed results of the selected discovery techniques can be found in Appendix A.

From Table 3.7, it can be concluded that there is much more significant difference amongst the process discovery techniques. ILP Miner clearly outperforms the other algorithms in terms of recall, whereas HeuristicsMiner has the upper hand with respect to precision. From these results, it becomes lucid why the use

	Recall			Precision			Run time
	f	r_B^p	*set*	$a_B^{'}$	s_B^n	p_B	
AGNEsMiner	*0,744*	*0,861*	*0,362*	0,590	0,897	*0,363*	10h:29m:2s
α^+	0,648	*0,551*	*0,033*	*0,491*	0,625	0,237	1s
α^{++}	*0,523*	*0,373*	*0,015*	*0,433*	0,624	0,249	1s
DT Genetic Miner	*0,751*	**0,871**	*0,383*	**0,697**	*0,896*	*0,329*	1d:6h:57m:47s
Genetic Miner	*0,524*	0,819	*0,227*	**0,713**	**0,935**	**0,429**	3d:12h:41m:53s
HeuristicsMiner	**0,775**	*0,729*	0,469	**0,778**	**0,960**	**0,516**	5s
ILP Miner	**0,965**	**0,951**	**1,000**	0,593	0,766	*0,249*	2m:8s
Flower	1,000	1,000	1,000	0,352	0,009	0,049	1s

Table 3.7: Average accuracy results for the real-life event logs

	Approach A				Approach B			
	r_B^p	p_B	$F1_{r_B^p,p_B}$	$F2_{r_B^p,p_B}$	f	$a_B^{'}$	$F1_{f,a_B^{'}}$	$F2_{f,a_B^{'}}$
AGNEsMiner	*0,861* (**3,4**)	*0,363* (**3,6**)	0,476 (**3,1**)	**0,612** (**2,8**)	*0,744* (**4,0**)	0,590 (5,1)	**0,642** (**4,3**)	**0,694** (**3,9**)
α^+	*0,551* (*5,4*)	0,237 (*5,7*)	*0,265* (*6,3*)	*0,338* (*6,6*)	0,648 (4,7)	*0,491* (4,8)	**0,547** (**4,9**)	*0,598* (**4,9**)
α^{++}	*0,373* (*6,6*)	0,249 (*5,3*)	*0,262* (*5,6*)	*0,296* (*6,3*)	*0,523* (*5,2*)	0,433 (*5,9*)	0,473 (**4,0**)	*0,501* (**5,1**)
DT Genetic Miner	**0,871** (**2,8**)	*0,329* (**4,0**)	0,457 (**3,9**)	**0,615** (**3,0**)	*0,751* (**3,9**)	**0,697** (**3,3**)	**0,703** (**4,6**)	**0,725** (**3,6**)
Genetic Miner	0,819 (**3,1**)	**0,429** (**3,0**)	**0,535** (**2,6**)	**0,652** (**2,5**)	*0,524* (*5,9*)	**0,713** (**3,0**)	*0,563* (**4,6**)	0,534 (5,5)
HeuristicsMiner	0,729 (5,0)	**0,516** (**1,9**)	**0,587** (**2,0**)	**0,656** (**2,6**)	**0,775** (**2,8**)	**0,778** (**1,8**)	**0,737** (**2,6**)	**0,753** (**2,5**)
ILP Miner	**0,951** (**1,8**)	*0,249* (**4,5**)	*0,371* (**4,5**)	*0,553* (**4,3**)	**0,965** (**1,6**)	*0,593* (**4,1**)	**0,699** (**3,0**)	**0,809** (**2,5**)
Flower	1,000	0,049	0,092	0,197	1,000	0,352	0,477	0,627

Table 3.8: Average F1- and F2-scores for the real-life event logs, with average ranks between brackets

of the F-score as an accuracy evaluation method becomes feasible. It provides a valuable approach to balance between recall and precision. By combining recall and precision values into one metric, the overall accuracy of discovered process models can be assessed.

The F-score results for the real-life event logs is shown in Table 3.8. From this table, it can be seen that HeuristicsMiner is the most powerful technique in a real-life environment. It outperforms the other algorithms for both evaluation approaches and according to both the parametric and non-parametric statistical evaluation. Only according to the $F2_{f,a_B^{'}}$ metric where recall is weighed twice as much as precision, ILP Miner shows better performance. However, this is mainly due to the fact that the precision metric ($a_B^{'}$) overestimates the actual precision of the model. For a detailed analysis on the drawbacks of $a_B^{'}$, we refer to the previous chapter. Another conclusion that can be formulated from Table 3.8 is that the α-algorithms are least suitable. This is not surprising because these algorithms are more theoretical in nature and therefore less robust in a real-life setting.

When comparing the performances on real-life event logs with those of the artificial logs, it becomes clear that there are significant differences between both. As for the artificial event logs, especially AGNEsMiner and the genetic algorithms rank at the top. However, for the real-life event logs, this picture shifts thoroughly as the HeuristicsMiner algorithm clearly outperforms these algorithms. What is also important to notice is the significant difference in average run time, which again favors the HeuristicsMiner algorithm.

Comprehensibility

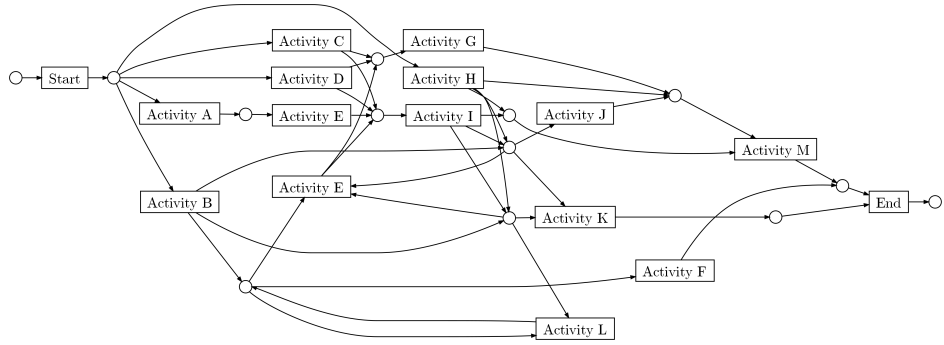
Next to accuracy, a second focus of this evaluation study is mined process model comprehensibility. The metrics identified for evaluating comprehensibility are described in Section 3.4.1. Despite the choice for quantifying comprehensibility by using process model complexity metrics, it goes without saying that assessing model comprehensibility in a quantitative way is challenging. Because comprehensibility remains largely subjective, relying on a visual analysis is definitely worthwhile in practice.

In order to clarify the discussion on comprehensibility to some extent, Figures 3.2, 3.3 and 3.4 show a selection of the discovered process models. Figures 3.2 and 3.3 show the discovery results of a less complex real-life event log (XAO). From this figure, it can be seen that even for relatively simple event logs, the comprehensibility of the models can differ significantly. For instance, the ILP Miner result shows a significantly higher number of arcs with respect to the number of places and transitions. What is more, the transformation of the models discovered by HeuristicsMiner and Genetic Miner (i.e. heuristic nets) to Petri nets entails the inclusion of different invisible transitions for routing purposes, which obviously decreases comprehensibility.

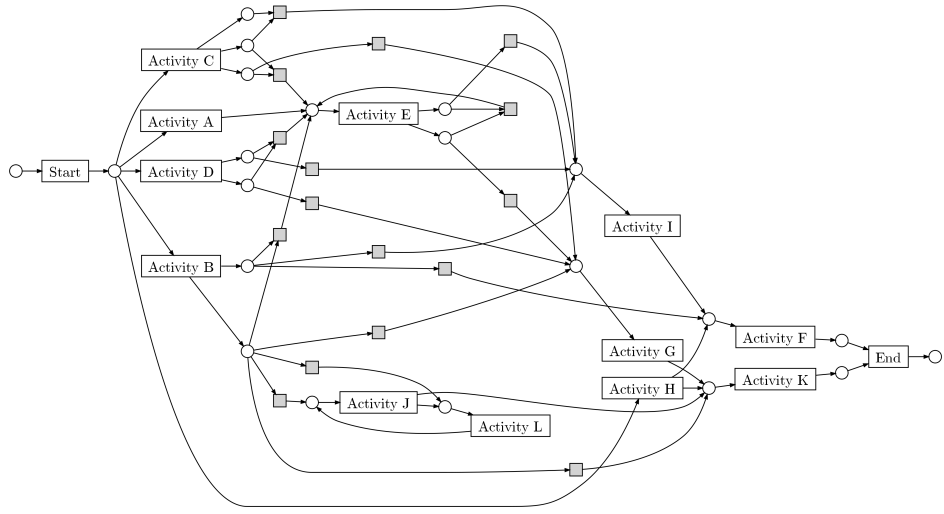
In contrast to the relatively understandable process models in Figures 3.2 and 3.3, the picture shifts when taking into account Figure 3.4. The fragment of the ILP Miner model clearly shows a severe degree of incomprehensibility. The HeuristicsMiner result might be judged more comprehensible to some extent, but overall comprehensibility is moderate as well. This indicates that for highly complex event logs, even better performing techniques such as HeuristicsMiner and AGNEsMiner fail to address the challenge of complexity. From the visual analysis, it can be concluded that foremost the number of arcs with respect to the number of nodes is an appropriate indicator of model comprehensibility.

As indicated earlier, a quantitative approach for assessing comprehensibility was opted for. Table 3.9 provides the average results over the eight real-life event logs for each of the algorithms. For those metrics that are not in the $[0, 1]$-interval, we scaled the results for each event log and afterwards averaged these scaled values. Notice that in this table the lower the value, the better. Furthermore, the average ranks are again reported between brackets. Note also that the ECyM metric could not be reported as an average because the metric resulted in a high amount of infinite values. The main reason hereto is the infeasibility to calculate the state space of a certain mined process model. Therefore, we did not include this metric in the evaluation table. Note that the log-specific comprehensibility results can be found in Appendix B.

The comprehensibility results do not allow for a straightforward interpretation. Therefore, we will discuss a number of key observations. First of all, Genetic Miner

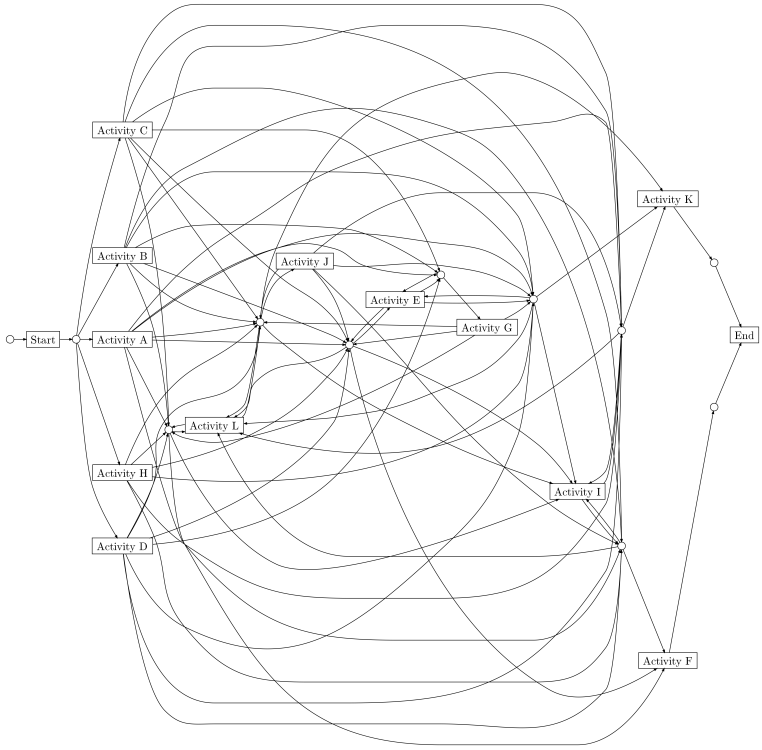


(a) AGNEsMiner

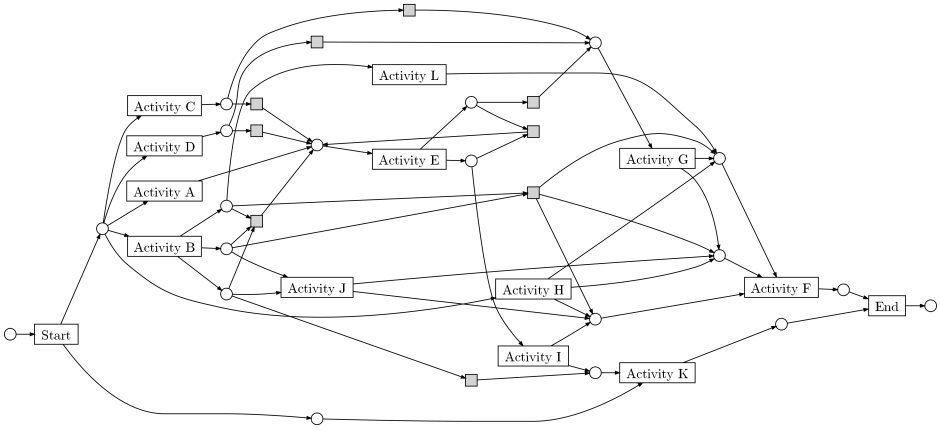


(b) GeneticMiner

Figure 3.2: Discovered process models by AGNEsMiner and Genetic Miner for the XAO data set

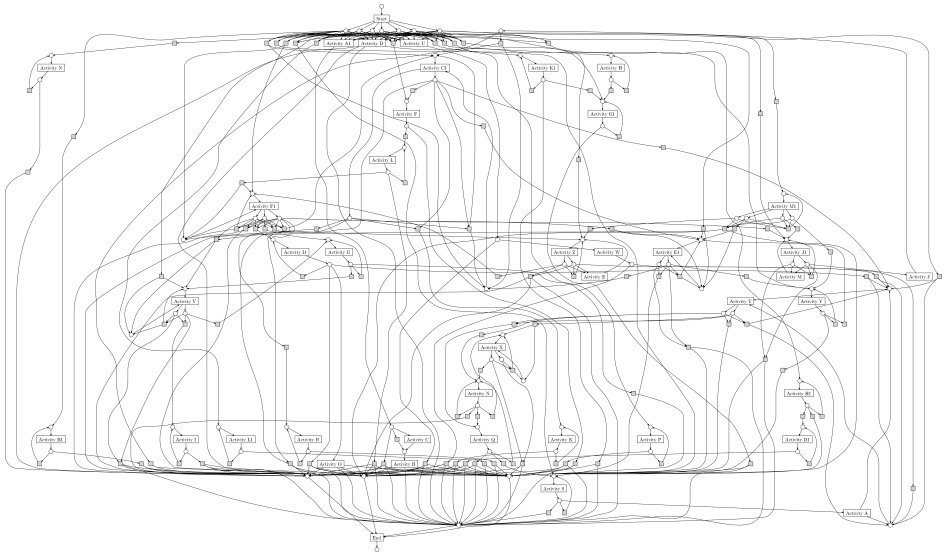


(a) ILPMiner

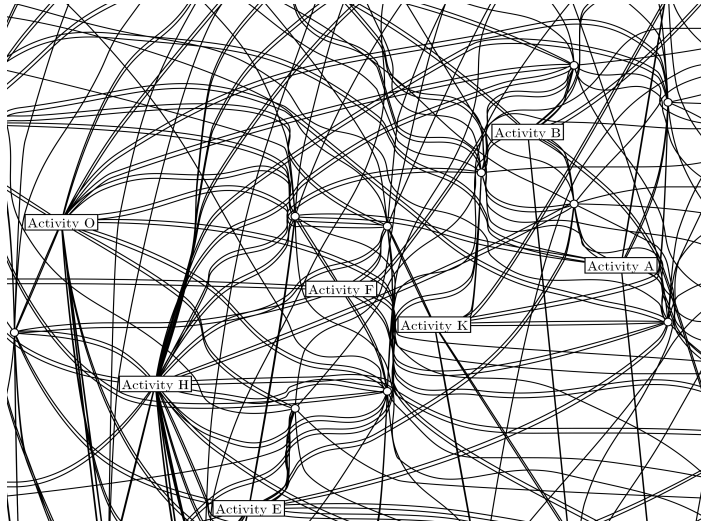


(b) HeuristicsMiner

Figure 3.3: Discovered process models by ILP Miner and HeuristicsMiner for the XAO data set



(a) Petri net discovered by HeuristicsMiner



(b) Fragment of the result of ILP Miner, clearly showing overspecification

Figure 3.4: Discovered process models for the complex UFM event log

	transi-tions	places	arcs	AND-Joins	AND-Splits	OR-Joins	OR-Splits	ECaM	SM
AGNEsMiner	*0,515* (**3,9**)	**0,332** (**2,4**)	**0,301** (**2,1**)	**0,402** (**2,9**)	**0,363** (**2,6**)	**0,267** (**2,6**)	**0,279** (**2,1**)	**0,275** (**2,5**)	**0,173** (**4,3**)
α^{+}	**0,354** (**2,6**)	**0,459** (**3,9**)	**0,367** (**3,3**)	**0,510** (**4,1**)	**0,515** (**4,2**)	**0,414** (**4,0**)	**0,374** (**3,8**)	**0,331** (**3,0**)	**0,257** (**3,2**)
α^{++}	**0,331** (**3,2**)	0,605 (**5,1**)	**0,641** (5,3)	**0,543** (**5,2**)	**0,633** (5,6)	0,757 (*6,1*)	0,710 (*5,6*)	**0,611** (**4,8**)	**0,240** (**4,8**)
DT Genetic Miner	*0,772* (5,3)	0,594 (**4,1**)	**0,400** (**3,0**)	**0,364** (**2,3**)	**0,392** (**2,6**)	**0,310** (**2,7**)	**0,309** (**2,6**)	**0,468** (**3,9**)	**0,245** (**3,6**)
Genetic Miner	*0,919* (*6,1*)	*0,834* (*6,1*)	*0,544* (**4,8**)	0,755 (**4,8**)	0,618 (**3,6**)	0,567 (**4,6**)	*0,656* (**4,9**)	*0,565* (**4,8**)	*0,585* (**5,3**)
HeuristicsMiner	*0,715* (5,2)	*0,482* (**3,9**)	**0,360** (**3,4**)	**0,633** (**4,4**)	*0,618* (**4,3**)	**0,364** (**3,0**)	0,410 (**3,4**)	0,372 (**3,8**)	0,209 (**4,5**)
ILP Miner	**0,381** (**1,9**)	**0,365** (**2,6**)	*0,866* (*6,1*)	**0,626** (**4,4**)	*0,742* (**5,1**)	*0,653* (**5,1**)	*0,696* (*5,7*)	*0,770* (**5,3**)	**0,140** (**2,4**)
Flower	0,409	0,046	0,141	0,000	0,000	0,031	0,036	0,021	0,002

Table 3.9: Comprehensibility results for the real-life event logs

is critically underperforming since all possible incomprehensibility indicators show that discovered process models by this algorithm are significantly more difficult to interpret than those of other process discovery algorithms. This adds to the fact that the algorithm is also problematic in terms of run time. Although Bratosin et al. [20] showed how the performance of genetic miner could be improved by applying a sampling procedure, we argue that the use of Genetic Miner in a real-life setting is inadvisable.

Secondly, regarding the Petri net building blocks, it can be seen that ILP Miner comes up with process models with an extremely high amount of arcs. More precisely, this algorithm creates on average 1633 arcs between transitions and places and vice versa. Whatever other metrics might indicate, this high amount of arcs renders any process model incomprehensible. As illustrated in Figures 3.2 and 3.3, even for easier data sets such as XAO, the overfitting behavior of ILP Miner clearly complicates comprehensibility.

Another important remark is the major contradiction between the ECaM-metric and the SM-metric. Although both metrics are not able to indicate comprehensibility differences between the techniques when taking into account the average ranks, the absolute values of these metrics dissent severely. This difference might be due to the discrepancy in local or global analysis of complexity. However, we think that it also indicates that it is not very straightforward to devise comprehensibility metrics that are suitable for process models mined from real-life event logs.

Notwithstanding the visual inspection of the discovered process models, the

results in Table 3.9 illustrate that it is challenging to prove significant differences between process discovery algorithms statistically. In addition, all of the presented metrics suffer from some kind of bias. Simple metrics such as number of places, arcs, AND-Joins/Splits and OR-Joins/Splits are influenced by the modeling language whereas the more advanced complexity metrics suffer from the difficulty to deal with highly unstructured process models which are less common within process modeling, the domain they were devised for. We can expect that by taking into account a multitude of metrics, these biases are slightly filtered out. Therefore, we think that this analysis remains valuable.

To conclude, it should be noted that in general, the comprehensibility of discovered process models is mediocre. Especially for the more difficult event logs like UFM and XNB, the discovered process models are highly unstructured and therefore uninterpretable, which shows that knowledge discovery from such data sets causes major problems for traditional process discovery techniques.

3.6.3 Multivariate Analysis

In this final part of the results section, we provide the outcome of the application of two statistical analysis techniques, as described in Section 3.5.3. These techniques allow for the formulation of some more general findings regarding the evaluation of process discovery techniques in a real-life setting.

Principal Component Analysis

Principal Component Analysis is a variable reduction technique that is well suited to analyze a number of interrelated variables. In this case, we applied PCA on a data set consisting of 61 observations, each observation being the results of one process discovery algorithm on one data set. Note that for three algorithm event log combinations, the discovery algorithm could not learn a process model from the data. Our main goal is to find out whether a high number of both accuracy and comprehensibility metrics can be reduced to a manageable few without much loss of information. The analysis took into account fifteen different metrics, as represented in Table 3.10.

Figure 3.5 is the scree plot, as proposed by Cattell [29], which allows to identify the number of principal components (PCs) to retain. The figure shows an elbow point at PC3. However, because the eigenvalue of the fourth principal component also exceeds 1, we report the principal component scores for the first four PCs. These four principal components account for 83% of the variance of the original data.

Table 3.10 shows the loadings of the different principal components with respect to the original variables. Firstly, it can be seen that the first principal component (PC1) is highly correlated with all the comprehensibility variables.

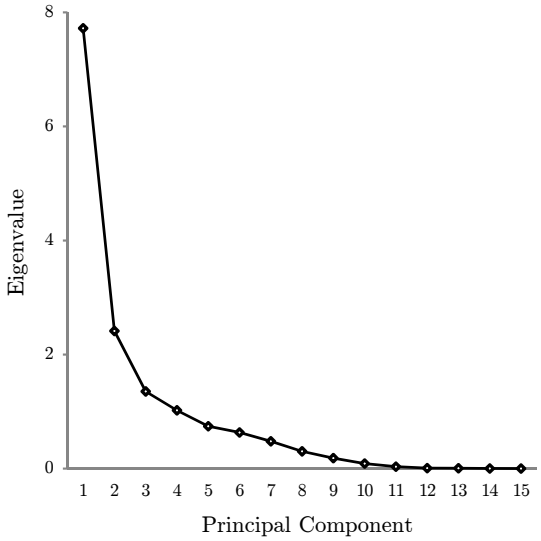


Figure 3.5: Scree plot of the PCA showing an elbow point at PC3

This PC can be seen as an aggregate of process model comprehensibility, which shows that the selected comprehensibility metrics correlate significantly. As such, by investigating the average principal component scores of PC1 for the different algorithms, we can conclude that Genetic Miner (2,39) and ILP Miner (0,58) can be considered more incomprehensible models whereas AGNEsMiner (-0,60) and HeuristicsMiner (-0,30) are better performing in this dimension. Table 3.10 further indicates that PC2 captures the variance in recall, whereas PC3 represents the precision of discovered process models. Finally, PC4 is strongly correlated with $a_S^{'}$.

In conclusion, this analysis implies that it can be useful to combine different metrics into a single variable in order to evaluate a process model in respect to a certain dimension. However, this combination of metrics is strictly different from the F-measure approach introduced in Section 3.1, which is a method to find a balance between different accuracy dimensions.

Canonical Correlations

Canonical correlation analysis is an appropriate statistical technique for analysis of relationships between two sets of variables. The technique, developed by Hotelling [73], accounts for the covariation of the variables from both within and across sets. Because we can expect to find correlations between accuracy, comprehensibility and log complexity, canonical correlation is a suitable method for analyzing the relationships between the dimensions proposed. The data set used consisted of

	PC1	PC2	PC3	PC4
$\mathbf{r_B^p}$	0,310	**0,695**	0,368	-0,168
f	**-0,515**	**0,707**	0,148	0,225
set	-0,206	**0,863**	0,169	0,112
$\mathbf{p_B}$	-0,077	-0,264	**0,762**	0,238
$\mathbf{a_B'}$	0,036	-0,494	**0,526**	0,403
$\mathbf{a_S'}$	0,112	0,070	-0,496	**0,808**
SM	**0,861**	-0,045	0,149	0,038
ECaM	**0,797**	0,404	-0,025	0,039
transitions	**0,816**	-0,078	-0,078	-0,196
places	**0,961**	-0,019	0,114	0,006
arcs	**0,832**	0,389	-0,027	0,032
AND-Joins	**0,956**	0,036	0,043	-0,062
AND-Splits	**0,959**	-0,011	0,036	-0,026
OR-Joins	**0,956**	0,117	0,079	0,074
OR-Splits	**0,944**	0,166	0,041	0,081

Table 3.10: Loadings of the principal components

the same 61 observations from the previous section extended with the event log characteristics described in Table 3.4.

The results of three separate canonical correlation analyses are presented, as shown in Table 3.11. These analyses differ from each other in terms of the choice of criterion and predictor variables. For every analysis, only the first canonical variate pair was significant at the 0,01 level, as determined by the F-value of the Wilks' Λ-tests. This test is equivalent to the likelihood ratio test and it verifies the null hypothesis that the canonical correlations in the current row and all that follow are zero. From this, we can conclude that the analyses allude to the presence of interdependence between the different sets of variables.

It should be noted that large canonical correlations may not imply strong correlation between the criterion and predictor variables. In order to discuss the practical significance of the canonical correlation analyses, we make use of a redundancy measure as proposed by Stewart and Love [122]. This measure determines how much of the variance in one set of variables is accounted for by the other set of variables. As such, the proportion of variance in the accuracy variables predictable from the log complexity variables was 21%. Similarly, the proportion of variance in the comprehensibility variables predictable from the log complexity variables amounted up to 24%. Finally, about 28% of the variance in the accuracy variables is shared by the comprehensibility variables. Taking into account that a lot of variance in the criterion variables is accounted for by the type of algorithm

	Criterion variables	Predictor variables	1st Can. Corr.
CCA 1	Accuracy	Log complexity	0,79
	f, r_B^p, set, $a_B^{'}$, p_B	# PI, # EV, # AT, # DPI, LoD, ST, MA	($F = 2,26$ with $p = 0,0002$)
CCA 2	Comprehensibility	Log complexity	0,75
	places, arcs, AND-Joins, AND-Splits, OR-Joins, OR-Splits, ECaM, SM	# PI, # EV, # AT, # DPI, LoD, ST, MA	($F = 1,63$ with $p = 0,0068$)
CCA 3	Accuracy	Comprehensibility	0,76
	f, r_B^p, set, $a_B^{'}$, p_B	transitions, places, arcs, AND-Joins, AND-Splits, OR-Joins, OR-Splits, ECaM, SM	($F = 1,86$ with $p = 0,0021$)

Table 3.11: Criterion and predictor variables for the three different canonical correlation analyses, with the value and the significance of the first canonical correlation

applied, we can conclude that there is practical evidence that log complexity has an important impact on both accuracy and comprehensibility. Furthermore, a noteworthy correlation exists between accuracy and comprehensibility.

To conclude this canonical correlation analysis, we provide the correlations between the first canonical variable and the original variables in Table 3.12. These correlations, also known as *structural correlations*, allow for interpreting the canonical variates. For example, from the first canonical correlation analysis (CCA 1), we can see that the first canonical variate represents foremost Behavioral Precision (p_B) and to a lesser extent fitness (f). Similarly, for the log complexity predictor variables, it can be seen that both the number of distinct process instances (# DPI) and the number of process instances (# PI) have an important influence on the first canonical variate LogComplexity1. Thus, the shared variance between accuracy and log complexity is mainly expressed by the influence of # DPI and # PI on p_B and f.

Furthermore, for CCA 2 it can be seen that the first canonical variate Comprehensibility1 is strongly correlated with ECaM, number of arcs and number of AND-Joins/Splits and LogComplexity1 is predominantly correlated with # AT. These positive correlations bring about that the number of activity types has a notable influence on process model comprehensibility.

Finally, ascertaining the relationship between accuracy and comprehensibility in CCA 3, it can be concluded that there exists a strong negative relation between predominantly the recall variables (f, r_B^p) and the number of transitions, the number of places and the number of AND-Joins/Splits. Accordingly, the shared variance between comprehensibility and accuracy is primarily channeled through

CCA 1		CCA 2		CCA 3	
Accuracy1		Comprehensibility1		Accuracy1	
f	-0,53	places	0,38	f	-0,88
r_B^p	-0,41	arcs	0,66	r_B^p	-0.63
set	-0,31	AND-Joins	0,64	set	-0,45
$a_B^{'}$	0,21	AND-Splits	0,64	$a_B^{'}$	0,27
p_B	-0,74	OR-Joins	0,38	p_B	-0,09
		OR-Splits	0,47		
		ECaM	0,68		
		SM	0,20		
LogComplexity1		LogComplexity1		Comprehensibility1	
# PI	0,55	# PI	0,44	transitions	0,73
# EV	0,30	# EV	0,29	places	0,69
# AT	0,21	# AT	0,94	arcs	0,36
# DPI	0,75	# DPI	0,42	AND-Joins	0,75
LoD	-0,10	LoD	-0,29	AND-Splits	0,72
ST	-0,44	ST	0,29	OR-Joins	0,59
MA	-0,35	MA	-0,55	OR-Splits	0,58
				ECaM	0,28
				SM	0,57

Table 3.12: Correlations between the first canonical variate and their original variables for each of the canonical correlation analyses (CCAs)

the recall variables.

3.7 Discussion

In Section 3.5, the quality assessment of process discovery techniques is restricted to algorithms that discover a Petri net or algorithms of which the resulting models can be transformed into a Petri net. There exist two important reasons why a Petri net-based evaluation methodology is opted for. Firstly, a majority of state-of-the-art discovery algorithms comply with this requirement. Furthermore, from an evaluation perspective, the majority of metrics for assessing discovered process models is defined in terms of Petri nets. In this way, a Petri net-based quality assessment is considered the most feasible approach to benchmarking process discovery algorithms.

3.7.1 Representational Bias

It is important to note that the choice for Petri nets as the representational language for discovered process models might collide with this study's main goal to assess process discovery techniques based on real-life event logs. For a more general discussion on the representational bias in process discovery, we refer to [134]. The main issue of our study is that especially for highly complex event logs

exhibiting a large variety in process behavior, relying on Petri nets as the representational language might be less suited. It has been found (e.g. [62, 133]) that Petri net- based discovery techniques sometimes lack the capabilities to learn accurate and comprehensible models from this type of event logs. In such cases, alternative discovery techniques such as the Fuzzy Miner [67], or event log manipulation techniques (e.g. pattern abstractions [14], sequence clustering [54, 66, 120], etc.) are considered more useful because these techniques feature better abstraction capabilities. Despite the potential of alternative discovery techniques with increased abstraction capabilities going beyond the discovery of a single process model, they require a different evaluation methodology since abstraction itself needs to be taken into account as an extra evaluation dimension. It should be noted that research with respect to holistically benchmarking different approaches for the analysis of highly complex event logs is to the best of our knowledge nonexistent in the process mining literature. The issue of representational bias is of course strongly related to the chosen evaluation methodology. Accordingly, the main limitations of our approach are discussed in the next section.

3.7.2 Limitations of a Petri Net-Based Evaluation Methodology

The limitations of the chosen representational bias, i.e. Petri nets, are twofold. Firstly, the representational bias might cause the inability to represent the underlying process model of a certain event log well. This is related to what is explained in the previous paragraph. Techniques that discover a single Petri net from an event log might be confronted with the problem that not even a single element in the search space is a satisfactory discovery result. As far as we know, no research has explicitly focused on the ability of different representational biases to discover process models from event logs, let alone on the improvement of the representational bias for process discovery. Notwithstanding that design-oriented languages such as Petri nets or EPCs are often used for the representation of discovered process models, the modeling origin often complicates the process discovery learning problem. As such, it should be investigated how to increase the flexibility of the representational bias, for instance by using less restrictive languages such as fuzzy nets [67], declarative models [138] or C-nets [135].

A second limitation following the choice for Petri nets is that discovered models can be internally inconsistent (i.e. not sound). This is because none of the discovery techniques limit the search space to sound workflow nets (WF-nets [124]). As such, discovered process models might contain livelocks and/or deadlocks. The problem of soundness is strongly confirmed by this study as well, since none of the discovery techniques was able to discover a sound Petri net, even from the more straightforward real-life event logs. Notwithstanding the fact that unsoundness

has a definite impact on both accuracy and comprehensibility of process discovery techniques, it remains debatable whether a Petri net-based process discovery technique's first priority should be to render sound process models. Soundness is foremost important in the process modeling domain, whereas the main goal of process discovery techniques is to provide insight into a set of process executions. Since soundness is not the top priority problem in process discovery, we find that currently available process discovery techniques apparently do not consider soundness in their learning strategies. It can be argued that the importance of soundness is such that it should be incorporated in the process discovery learning problem. However, it is shown that the process discovery learning problem is highly challenging even without considering soundness. Accordingly, it can be advocated that due to low-level nature and additional criteria such as soundness, Petri nets might not be the most suitable representational bias for process discovery.

Despite the limitations of Petri net-based discovery techniques, it is beyond the scope of this study to explore or assess the whole aspect of the representational bias in process discovery. Furthermore, it is rarely known upfront whether an event log will create major challenges for traditional process discovery techniques. As such, it remains very useful to know which discovery techniques perform better on real-life event logs, even if after initial analysis improved abstraction techniques may provide more useful results. In addition, many solutions for the complexity problem arising from event logs displaying a high variety of process behavior, will inevitably make use of some sort of process notation. In spite of the shortcomings, Petri nets remain a valid option for this task.

3.7.3 The Issue of Parameter Tuning

As described in Section 3.5.1, we opted to employ standard parameter settings except for those techniques for which there exists sources in the literature to alter these standard parameters. For instance, for Genetic Miner and AGNEsMiner, we adopted the parameters according to the demonstrated settings in [10] and [63] respectively. It is acknowledged that this experimental setup choice restricts the level of generality of our conclusions. Nonetheless, parameter tuning of process discovery techniques is very complicated. As illustrated in the results section, many techniques require multiple hours of run time before presenting a result. Furthermore, the amount of time required to calculate the evaluation metrics is sometimes even larger. Accordingly, complete parameter tuning of all assessed techniques is totally infeasible. Also in [23], parameter tuning of the HeuristicsMiner algorithm is investigated. Due to the real-valued parameters, a designated solution strategy is proposed, which relies on discretizing real-valued parameters and applying the Minimum Description Length (MDL) principle to calculate the

optimal solution. Nevertheless, the use of this kind of strategy for the whole set of assessed techniques would require huge computation times due to computational complexity of both the model building as well as the calculation of evaluation metrics. As such, this strategy is deemed infeasible.

To conclude, developers of process discovery techniques should be aware of the fact that extensive parameter tuning is often completely infeasible. Therefore, they should either design algorithms which require few parameters or they should validate their novel technique on wide ranges of data sets in order to define ranges for the required parameters which can serve as guidelines for application of the technique in practice. In the domain of data mining, Kriegel et al. [79] pointed out that designers of data mining algorithms should avoid the use of type-iparameters so as to increase the usability of the algorithm. They also state that if type-iparameters are required, it is a best practice to try to automatically configure the parameters by including the parameter tuning in the optimization problem.

3.8 Conclusion

Assessing the quality of process discovery techniques is an essential element for both process mining research as well as for the use of process mining in practice. Up till now, evaluation of process discovery techniques was performed in very diverse ways, but mostly on artificial data sets. In this study, a multi-dimensional evaluation study was carried out in order to evaluate process mining techniques comprehensively on eight real-life event logs. The results of our study allow for the formulation of a number of key conclusions:

- There exists an important difference between evaluation of process discovery algorithms based on either artificial or real-life event logs. The use of real-life event logs for process discovery quality assessment is a major contribution of this study.
- The F-score methodology is a valuable approach to combine recall and precision metrics in order to assess overall accuracy of a process model.
- HeuristicsMiner seems the most appropriate and robust technique in a real-life context in terms of accuracy, comprehensibility, and scalability.
- It was statistically validated that dealing with high complexity event logs remains an important challenge for process mining algorithms, both in terms of accuracy as well as in terms of comprehensibility.
- Proficient quantification of discovered process model comprehensibility still faces fundamental challenges. Metrics from the process modeling domain

were applied in this study, but were found to be difficult to interpret because of the low-level of structure many mined, real-life process models exhibit.

- Multivariate statistical analyses showed that it might be useful to reduce different metrics into one single quantification of a certain dimension. Furthermore, it was demonstrated that there exists important correlations between process model accuracy, process model comprehensibility and event log complexity.

- Representational bias is a widely overlooked research topic in the process mining domain. This study centering on single model Petri net discovery techniques shows that for complex real-life event logs, alternative analysis techniques can be expected to be an interesting follow-up analysis.

Overall, we think that the future of process mining research should emphasize on developing insightful techniques for analyzing real-life event logs. Although currently available control-flow discovery techniques can still play a significant role in conducting process mining analysis in practice, more complex event logs require novel data exploration techniques with a definite focus on comprehensibility. For instance, Bose and van der Aalst [18] show how trace alignment might be used to derive insights from complex event logs. In similar fashion, trace clustering is also an interesting process data exploration technique offering a solution for highly complex data sets. Therefore, in the next chapter, a novel trace clustering technique is proposed that provides an alternative methodology for solving the learning difficulties that many single model process discovery techniques are confronted with.

Chapter 4

Active Trace Clustering for Improved Process Discovery

The previous chapter clearly indicated that the simple application of existing, single model process discovery techniques will often yield highly incomprehensible process models because of their inaccuracy and complexity. With respect to resolving this problem, *trace clustering* is one very interesting approach since it allows to split up an existing event log so as to facilitate the knowledge discovery process. In this chapter, a novel trace clustering technique is described which significantly differs from previous approaches. Above all, it starts from the observation that currently available techniques suffer from a large divergence between the clustering bias and the evaluation bias. By employing an active learning inspired approach, this bias divergence is solved. In an assessment using both a controlled environment as well as four real-life event logs, it is shown that our technique significantly outperforms currently available trace clustering techniques from a process discovery evaluation perspective.

4.1 Trace Clustering in Process Mining

Process Mining has been demonstrated to possess the capabilities to profoundly assess business processes [135]. In particular, process mining techniques are highly suitable in flexible environments such as healthcare, customer relationship management (CRM), product development, etc. [68]. This is because information systems in such environments often grant a higher degree of freedom to their users. Accordingly, Process Mining proves to be valuable by discovering the actual process at hand.

The starting point of analysis is an *event log*, which is basically a set of process executions capturing the different business activities that were performed in the context of a certain case. In this study, the control-flow perspective, i.e. the different transition relationships between the activities in the event log, is the object of analysis. However, typical event logs will contain much more information, for instance organizational information concerning the performers of the different activities [121].

The most crucial learning task in the process mining domain is termed *process discovery* and can be defined as the construction of a process model from an event log [118, 131]. Process discovery is a largely unsupervised learning task in nature due to the fact that event logs rarely contain negative events to record that a particular activity could not have taken place. Despite the demonstrated usefulness in flexible environments, it has been shown that *process discovery* is most challenging in this context. Various studies illustrate that process discovery techniques experience difficulties to render accurate and interpretable process models out of event logs stemming from highly flexible environments [15, 62, 68, 150]. This problem is largely due to the high variety of behavior that is captured in certain event logs. Accordingly, different approaches have been proposed to cope with this issue. Next to event log filtering, event log transformation [14] and tailor-made discovery techniques such as Fuzzy Mining [67], *trace clustering* can be considered a versatile solution for reducing the complexity of the learning task at hand. This is because one can rely on multiple process models to represent the variety in behavior of a certain event log by separating execution traces into different groups. The purpose of clustering process executions is detailed in Figure 4.1. By dividing traces into different groups, process discovery techniques can be applied on subsets of behavior and thus improve the accuracy and comprehensibility.

In this chapter, we build further on the idea of clustering event log traces in order to reduce the complexity of the process discovery learning task. Therefore, a novel approach is described that aims at improving currently available techniques by directly optimizing the accuracy of the cluster's underlying process models. In this way, the gap between the clustering bias and the evaluation bias from which currently available techniques suffer, is bridged. As such, this chapter is structured as follows. In Section 5.3.2 we provide an overview of existing approaches to trace clustering. Section 4.3 details the new trace clustering approach. Then Section 4.4 introduces multiple metrics for evaluating the quality of a trace clustering technique. These metrics are employed in Section 4.5 in which the novel trace clustering technique is assessed and compared to existing techniques, both in a controlled environment as well as by making use of 4 real-life event logs. Finally, Section 4.6 provides a discussion before conclusions are formulated in Section 5.1.4.

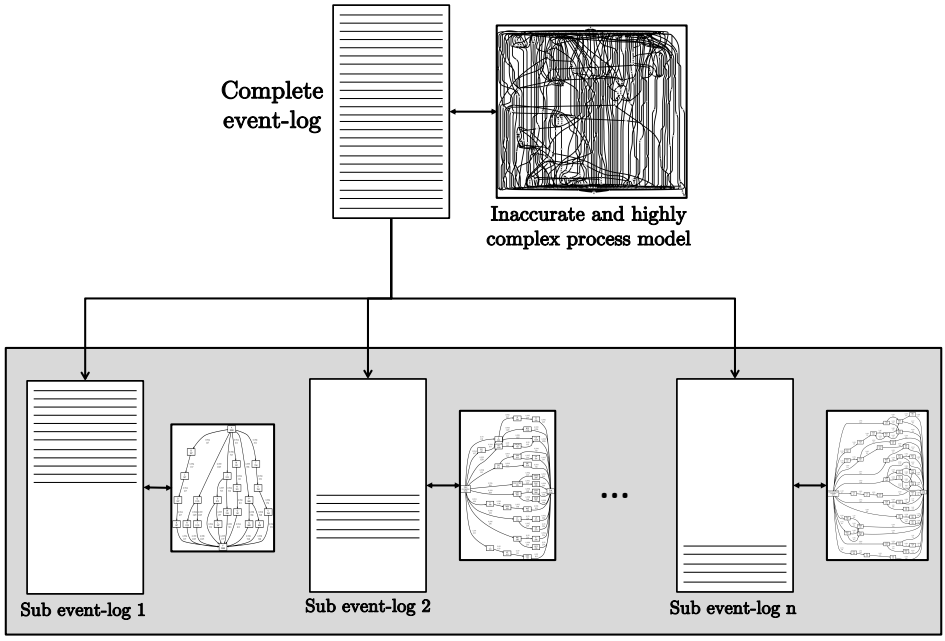


Figure 4.1: Illustration of the purpose of trace clustering in Process Mining

4.2 Related work

In the literature, different approaches to trace clustering have been proposed. Many techniques apply a kind of translation to the learning problem so as to make use of the existing distance-based clustering algorithms proposed in the data mining domain. For instance, by converting an event log into a vector space model, a distance metric can be defined between each couple of traces. However, there also exist techniques that define the distance between two traces without the translation to an attribute-value context. Furthermore, model-based clustering techniques were shown to be applicable for trace clustering as well.

Vector space approaches Greco et al. [66] were pioneers in studying the clustering of log traces within the process mining domain. They use a vector space model considering the activities and the activity transitions to cluster the traces in an event log with the purpose of discovering more simple process models for the subgroups. The authors propose the use of Disjunctive Workflow Schemas (DWS) for discovering process models. The underlying clustering methodology is k-means clustering. Song et al. [120] elaborate on the idea of constructing a vector space model for traces in an event log. In contrast to [66], this technique allows for a multitude of so-called *profiles* to determine the vector associated with each

process instance. As such, they define an activity profile, transition profile, performance profile, case attribute profile, etc. Furthermore, the implementation of the technique presents a full range of distance metrics and clustering techniques.

Context-aware trace clustering The most recent trace clustering techniques in Process Mining are described by Jagadeesh Chandra Bose and van der Aalst [15], [16]. The clustering techniques presented in these studies extend contemporary approaches by improving the way in which control-flow context information is taken into account. In [15], the authors propose a generic edit distance technique which is founded on the Levenshtein distance [85], a distance metric for comparing strings. The approach relies on deriving specific substitution, insertion and deletion costs so as to take into account the behavior in the event log. The idea of context-aware trace clustering is further developed in [16]. In this study, the authors return to the principle of generating a vector space model for the traces in an event log. However, instead of using activities and/or transitions as a basis for the vector space model, it is proposed to use conserved patterns or subsequences. In this way, the authors define Maximal, Super Maximal, and Near Super Maximal Repeats to create feature sets that determine the vector of a certain trace. Note that one of the advantages of the pattern-based feature sets over the generic edit distance approach is that it can be combined with feature sets that rely on other types of information as in [120].

Model-based sequence clustering A totally different approach to trace clustering was proposed by Ferreira et al. [55]. Inspired by the work of Cadez et al. [24] in the area of web usage mining, they propose to cluster sequences by learning a mixture of first-order Markov models using the Expectation-Maximization (EM) algorithm. In [150], this model-based trace clustering technique is applied to server logs with the purpose of demonstrating its usefulness in a real-life setting. Note that the technique presented in the next section is most similar to [55], notwithstanding that the underlying model representation and the clustering strategy is different.

4.3 Active Trace Clustering

This section introduces the new trace clustering approach. Its conception is significantly different from earlier techniques because it builds upon the observation that traditional trace clustering techniques suffer from a divergence between the clustering bias and the evaluation bias.

4.3.1 Clustering Bias vs. Evaluation Bias

The key idea behind existing trace clustering algorithms is to group traces that exhibit similar features. Although this idea is intuitive and has proven useful, it suffers from a fundamental problem. In a traditional data mining setting, clusters are created and evaluated based upon the idea of maximizing intra-cluster similarity and minimizing inter-cluster similarity. As such, currently available clustering techniques apply either a hierarchical or a partitional method like k-means for creating clusters. However, as described in [16], the most important evaluation dimension for trace clustering is from a process discovery perspective. This entails that for each cluster, a certain discovery technique will create a process model and the corresponding process model accuracy and process model complexity are combined to determine the quality of a certain clustering solution. Because existing techniques do not take into account the quality of the underlying process models of each of the clusters during the clustering procedure, these techniques suffer from a strong divergence between the evaluation bias and the clustering bias. Accordingly, it is highly questionable whether the existing clustering approaches will yield satisfactory results in terms of process models. In Section 4.5, it will be shown that this bias divergence indeed causes inferior results for existing trace clustering approaches.

Against this background, this chapter puts forward an entirely different approach to trace clustering. It does not rely on a vector space model, nor does it define a metric for quantifying similarity between two process instances. Instead, the technique proposed in this study is designed so as to solve the problem of finding an optimal distribution of execution traces over a given number of clusters whereby the combined accuracy of the associated process models is maximized. Note that brute-force solution strategies for this optimization problem are computationally intractable because they scale exponentially in terms of the number of distinct process instances in the log. For instance, in case of only 10 distinct traces and 4 clusters, more than 1 million different solutions are possible. Therefore, our technique proposes a top-down, greedy algorithm that computes a solution for this problem by grouping traces not because they exhibit similar behavior but because they fit well in a certain process model.

4.3.2 An Active Learning Inspired Approach

We term the novel clustering technique *Active Trace Clustering (ActiTraC)*. It is called *active* since it is inspired by a couple of principles of active learning in the field of machine learning. For a detailed survey of active learning literature, we refer to Settles [117]. The key idea behind active learning is that a machine learning algorithm can achieve better results with fewer training data if it is allowed to choose the data from which it learns [11], [31]. Despite the fact that active

learning is generally employed in the context of supervised learning techniques in order to reduce the classification error or label uncertainty (e.g. [90]), studies like [72] have demonstrated that the principles of active learning can be employed in an *unsupervised* setting as well. In this way, our proposed trace clustering technique borrows a couple of elements from the general idea of active learning. Most importantly, a *selective sampling* strategy is employed. Typically, an active learner will decide upon which instances to select based on some informativeness measure. In this case, the frequency of a trace is used as the primordial measure for its informativeness. Furthermore, the proposed technique is a greedy approach which does not guarantee an optimal solution in terms of process model accuracy. Finally, the clustering solution will often be based on a subset of the event log traces. Nonetheless, it is recognized that there exist differences as well. For instance, active learning in a supervised context focuses on problem areas for which the current model is uncertain. Mostly due to the fact that our approach is situated in an unsupervised context, our technique does not really focus on problem areas.

4.3.3 Notation

Before outlining the ActiTraC algorithm, some important concepts and notations that are used in the remainder of this study are discussed. An event log (L) consists of events (ev) that pertain to process instances. A process instance (pi) is defined as a logical grouping of activities whose state changes are recorded as events. As such, it is assumed that it is possible to record events such that each event refers to an activity type (at), i.e. a step in a process instance, a case, i.e. the process instance, and that events are totally ordered. Note also that identical process instances (i.e. traces with the same sequence of events) can be grouped into a distinct process instance, further denotes as *dpi*. A *dpi* is defined as a set of *pi* with the number of *pi* in the set denoted as the frequency of the *dpi*. A collection of *dpi*'s is called a grouped event log (GL).

4.3.4 A Three-Phase Algorithm

In general, Active Trace Clustering (ActiTraC) aims at creating clusters of event log traces for which the resulting process model is accurate according to some metric. The algorithm consists of three distinct phases: *selection*, *look ahead* and *residual trace resolution*. Its pseudo-code is depicted in Algorithm 1. Before the actual three phases of the algorithm can be carried out, identical process instances *pi* in event log L are grouped into distinct process instances *dpi*. Furthermore, the set of created clusters CS is initialized to the empty set and the set of remaining distinct process instances R contains all *dpi*'s created in the first step.

Algorithm 1 ActiTraC

Input: An event log L, the number of clusters nb_{clus}, a target fitness tf, a minimum cluster size mcs and a window size w
Output: A collection of event logs, represented by a set of clusters CS

Convert L into a grouped event log GL
$CS \leftarrow \emptyset$ # CS *denotes the set of clusters*
$R \leftarrow GL$ # R *denotes the set of remaining dpi's*
while $(|CS| < nb_{clus}) \wedge (R \neq \emptyset)$ **do**
  $C \leftarrow \emptyset$ # C *denotes the set of dpi's in the cluster*
  $I \leftarrow \emptyset$ # I *denotes the set of ignored dpi's*

Phase 1: Selection

  repeat
    Define W as the union of the most frequent dpi in $R \setminus I$ and the set of the top w% dpi's in $R \setminus I$ according to frequency # w *specifies the size of the window*
    if $C = \emptyset \vee |W| = 1$ **then**
      $cur\_dpi \in R | (\forall dpi \in R : |dpi| \leq |cur\_dpi|)$
    else
      $cur\_dpi \in W | (\forall dpi \in W : dist_{\text{MRA}}(C, dpi) \geq dist_{\text{MRA}}(C, cur\_dpi))$ # *function* dist_{MRA} *is the* ***average*** *MRA-based Euclidean distance between the dpi's in* C *and* cur_dpi
    end if
    $PM \leftarrow HM(C \cup \{cur\_dpi\})$
    if $fitness(PM) \geq tf$ **then**
      $C \leftarrow C \cup \{cur\_dpi\}$
      $R \leftarrow R \setminus \{cur\_dpi\}$
    else
      if $\sum_{dpi \in C} |dpi| \geq mcs \times \sum_{dpi \in R} |dpi|$ **then**
        $PM \leftarrow HM(C)$

Phase 2: Look ahead

        for all $dpi \in R$ **do**
          if $fits(dpi, PM)$ **then**
            $C \leftarrow C \cup \{dpi\}$
            $R \leftarrow R \setminus \{dpi\}$
          end if
        end for
        exit repeat

      else
        $I \leftarrow I \cup \{cur\_dpi\}$
      end if
    end if
  until $(R = \emptyset) \vee (R = I)$
  add the constructed cluster C to CS
end while

Phase 3: Residual trace resolution

if create new cluster for remaining traces **then**
  add new cluster C to CS with $C \leftarrow R$
else
  add each dpi in R to that C in CS for which its fitness, as calculated on the underlying process model, is highest
end if

Selection

In the first phase, traces are iteratively selected using a selective sampling strategy. The goal is to add a new distinct process instance to the set of already selected instances, with the purpose of evaluating the process model discovered from this new sub-event log. If the process model remains accurate enough, the selected trace is added to the current cluster and the selection procedure is repeated. As long as it is possible to add a trace to the current cluster without decreasing the process model accuracy, this process continues. To deal with the problem that small clusters are created, the *minimum cluster size (mcs)* parameter, ranging between 0 and 1, can be used so as to continue the selection and model building phase when an instance is encountered that results in an unfit process model (line 20). In this way, the algorithm can skip certain traces to increase the size of the current cluster.

Frequency-based selective sampling. The *window size (w)* parameter in line 9 gives rise to two variants of the ActiTraC algorithm. In case w is set to 0, the frequency window W will only contain the most frequent *dpi* in R. In this case, a straightforward selective sampling strategy is put in place that beholds the selection of the most frequent *dpi*. For this selected *dpi* and all the *dpi*'s already in C (with C denoting the set of *dpi*'s being part of the current cluster), a process model (PM) is calculated based on the function *HM*. *HM* denotes the application of the Heuristics Miner algorithm to the set of *dpi*'s. In the remaining part, this frequency-based selective sampling algorithm will be denoted ActiTraC$^{\text{freq}}$.

Distance-based selective sampling. Next to the basic frequency-based sampling, ActiTraC provides the flexibility to include a more complex selection strategy. In fact, ActiTraC can be adapted so as to take into account any kind of distance function between distinct process instances. As such, by increasing the window size w, it becomes possible to define a selective sampling strategy that incorporates the idea of clustering more similar traces together. In this study, it is opted to include an MRA-based Euclidean distance function as defined in [16]. This ActiTraC variant is further denoted as ActiTraC$^{\text{MRA}}$. The MRA-technique was selected because of its computational tractability and because it was found better performing in [16]. Note that the distance function in line 13 of Algorithm 1 is applied as follows: from the current set of remaining *dpi*'s, a window is defined based on frequency. For example, the 25% most frequent *dpi* form the window. Within this window, the algorithm targets that *dpi* with the lowest *average* MRA-based Euclidean distance with respect to the current set of *dpi*'s in C. In this way, more similar traces in terms of control-flow behavior are tested first and the resulting clustering solution will significantly differ from the first approach

solely relying on frequency. Note that it was opted to rely on frequency partly by defining a window in order to guide the clustering process towards creating a sufficiently large cluster with respect to the amount of remaining traces. In a similar way as with frequency-based selective sampling, the *dpi* is added to the current cluster and a process model (PM) is calculated with HeuristicsMiner.

Once this process model is created, it is verified whether the fitness of the process model is still above a predefined threshold, the *target fitness (tf)*. If the fitness remains higher than the *tf*, the *dpi* is added to the current set of instances C and a new selection is initiated. However, if the process model fitness drops below the *tf*, it should be verified whether the *minimum cluster size (mcs)* is attained. If so, selection is halted and phase two commences. Nevertheless, if the *mcs* is not met, the currently selected *dpi* is added to a set of skipped *dpi*'s I and the selective sampling of a new trace from R continues.

Look ahead

Once a process instance is encountered that decreases the model's accuracy below the specified threshold of the current cluster and the minimum cluster size is reached, the first phase comes to an end. In the second, *look ahead* phase, the remaining instances in the event log are taken into consideration (i.e. those traces that have not been subject to the selection phase yet) by verifying whether some of these traces do fit the process model created in the first phase. In this forward-looking procedure, only distinct process instances that fit the current process model perfectly (i.e. instances with an individual fitness of 1), are added to the current cluster. Instances that do not fit the model remain in the event log for which the selection phase can be started again to create a second cluster. This iteration of selection and look ahead is continued until the predefined maximum number of clusters is reached.

Residual trace resolution

Finally, the third phase specifies the resolution of the remaining traces in the event log. Either the remaining instances can be separated into a distinct cluster or these traces can be distributed over the created clusters according to the individual trace fitness for the different process models created. Note that in the remainder, only the latter option will be investigated because this choice will prevent the creation of strongly skewed clusters in terms of their size, which might in its turn trick the evaluation criteria as presented in Section 4.4.

4.3.5 Assumptions

The presented algorithms can deal with most event logs. There exists one major assumption regarding the frequency distribution of the process executions. One of the underlying principles is to prioritize traces with higher frequency over traces with a lower frequency. Of course, this can only be achieved in case of a non-uniform distribution of trace frequencies. It can be assumed that in a majority of real-life event logs, a non-uniform distribution is observed. However, it is pointed out that for event logs with a close to uniform distribution, the algorithm will present a result, but it will be challenging to attain sufficiently large clusters in the first two phases of the algorithm.

Further, the algorithm is also subject to the implementation of a mining technique and an accuracy evaluation measure. Currently, it was opted to use HeuristicsMiner [153] as the underlying discovery technique. This is because HeuristicsMiner has been demonstrated to possess the best capabilities to deal with real-life event logs in Chapter 3. The low computational costs of HeuristicsMiner as compared to other available techniques is an essential requirement to keep control over the computational demands of ActiTraC. Finally, ActiTraC relies on the ICS-fitness [152] for both the calculation of the overall accuracy of a process model as well as for the calculation of the individual fitness of a single process instance. This metric is similar to the well-known fitness metric defined by Rozinat and van der Aalst [107], even though it is a metric for heuristic nets. Note that next to the scalability of the metric's replay procedure due to the avoidance of costly state space calculations, another advantage of ICS-fitness is that it is not a pure recall metric since it also punishes overly general models.

4.4 Cluster Quality Criteria

The evaluation of a trace clustering solution can be approached from two perspectives. On the one hand, trace clusters can be assessed based on whether they group traces that present domain-specific similarities. On the other hand, a set of trace clusters can be evaluated from a process mining perspective, judging whether the technique achieves the goal to create more accurate and more comprehensible process models. In the next sections, both evaluation dimensions will be detailed.

4.4.1 Domain-Based Evaluation

Trace clustering techniques can have a primary objective to find groups of traces that conform to some domain-related criterion. For instance, in a healthcare context, a particular diagnosis will determine the specificities of the according care flow pathway.

Domain-related similarities between execution traces manifest themselves predominantly in resource information and case data. Nevertheless, trace clustering techniques typically rely on control-flow information only so as to cluster event log traces. This might strongly interfere with the objective to find domain-relevant clusters because it cannot be guaranteed that control-flow information only suffices. As such, finding those control-flow characteristics being the drivers behind domain-relevant clusters seems like finding a needle in a haystack.

Entropy

Quantifying the quality of a trace clustering technique from a domain perspective suffers from an important drawback which originates in the unsupervised nature of the learning problem. Without any ex ante information about how many domains should be found and which execution traces belong to which domain, it is unfeasible to construct domain-based evaluation metrics. In case there is ex ante information available, for instance in a gold standard evaluation setting, it is possible to define evaluation metrics that quantify how well a certain technique distinguishes the different classes of behavior. Examples of such metrics are true positives, false positives, purity, etc. Another powerful metric that can quantify clustering quality in a supervised setting is entropy. Entropy, a measure of disorder, has its origins in information theory, but can be applied advantageously in a clustering setting. For the purpose of this study, the entropy of a set of trace clusters is defined as follows.

Cluster Set Entropy (H_{CS}): Let k denote the number of clusters, q the number of classes, n the number of pi's in the event log, n_i the number of pi's in cluster i, and n_{ij} the number of pi's in cluster i belonging to class j. Then H_{CS} is defined as follows:

$$H_{CS} = -\sum_{i=1}^{k} \left(\frac{n_i}{n} \sum_{j=1}^{q} \left(\frac{n_{ij}}{n_i} \; log_2 \frac{n_{ij}}{n_i} \right) \right) \tag{4.1}$$

4.4.2 Process Mining-Based Evaluation

Next to domain, an even more crucial evaluation perspective is from a process mining viewpoint. With the goal of any trace clustering technique being to improve the accuracy and complexity of the process models discovered from an event log, both accuracy and complexity should be included in a proper quantification of the quality of a trace clustering solution. Note that this evaluation approach is largely inspired by the ideas put forward in [16].

Process model accuracy

Process model accuracy quantifies how well the observed behavior in an event log is captured by the discovered process model. In the process mining literature, assessing the quality of a process model with respect to a corresponding event log is termed conformance checking [107], with *trace replay* an often employed technique to calculate metrics. In Chapter 2, we have argued that assessing the accuracy of a discovered process model ideally involves an evaluation of multiple dimensions of which recall and precision are to be considered most important. In this way, we make use of our F1-score so as to assess the overall accuracy of the underlying discovered process model.

Obviously, individual F-scores for each of the clusters should be aggregated to assess a clustering solution as a whole. In order to take into account differences in size between the clusters, it was opted to weigh the individual F-scores based on the number of traces (pi) in each cluster. Note that the F-measure also allows to weigh recall and precision differently. In this case, it was opted to weigh them equally. As such, the Weighted Average F1-score is defined as follows:

Weighted Average F1-score ($\mathbf{F1}_{W.A.}$): Let n_i denote the number of traces in cluster i ($1 \leq i \leq k$), $\mathrm{F1}_i$ is the F1-score of the process model for trace cluster i. Then $\mathrm{F1}_{W.A.}$ is defined as follows:

$$\mathrm{F1}_{W.A.} = \frac{\sum_{i=1}^{k} n_i \mathrm{F1}_i}{\sum_{i=1}^{k} n_i} \tag{4.2}$$

Note that due to the incorporation of both recall and precision, the Weighted Average F1-score will punish process models that do not represent the behavior in the event log very well, as well as process models which over-generalize the behavior.

Process model complexity

Next to improving process model accuracy, the goal of trace clustering is to render more comprehensible process models. This comprehensibility can be quantified using process model complexity metrics.

In the literature, Lassen and van der Aalst [82] describe a number of metrics for quantifying process model complexity: Extended Cardoso metric (ECaM), Extended Cyclomatic metric (ECyM) and Structuredness Metric (SM). The main issue with these metrics is that they are designed from a process modeling perspective and thus assume save and sound workflow nets (WF-nets). Since process

discovery techniques cannot guarantee the discovery of nets that comply with these properties, it was opted to define a more straightforward metric.

The definition of this metric is supported by the work of Mendling et al. [94] who perform a study on the influence factors of process model understandability. A main result of this study is that very simple metrics such as the number of elements, the number of arcs, etc. are relatively good quantifiers of process model complexity. Based on the assumption that the process models underlying the formed clusters are represented by Petri nets [98], we define the Place/Transition Connection Degree as follows:

Place/Transition Connection Degree (P/T-CD): Let $|a|$ represent the number of arcs in a Petri net, $|P|$ the number of places and $|T|$ the number of transitions. Place/Transition Connection Degree P/T-CD is defined as follows:

$$\text{P/T-CD} = \frac{1}{2}\frac{|a|}{|P|} + \frac{1}{2}\frac{|a|}{|T|}. \tag{4.3}$$

In other words, P/T-CD is the weighted sum of the average number of arcs per transition and the average number of arcs per place. It is argued that the complexity of each of the models adds equally towards the difficulty of interpreting the entire clustering solution and therefore it is opted to not weigh the model complexity according to the sizes of the cluster. Accordingly, the Average Place/Transition Connection Degree is defined as follows:

Average Place/Transition Connection Degree (P/T-CD_A): Let k denote the number of clusters in the clustering solution, and P/T-CD_i the Place/Transition Connection Degree of the process model of cluster i. Then P/T-CD_A is defined as follows:

$$\text{P/T-CD}_A = \frac{\sum_{i=1}^{k} \text{P/T-CD}_i}{k}. \tag{4.4}$$

It is argued that this connection degree metric is an appropriate quantifier of process model complexity, especially for the purpose of evaluating trace clustering techniques because incomprehensible process models are often characterized by a drastic increase in the number of arcs that connect places and transitions.

Inter- and intra-cluster similarity

A standard approach to evaluate clustering techniques in the data mining domain relies on the paradigm to maximize intra-cluster similarity while minimizing inter-cluster similarity. In a supervised trace clustering setting where the classes of behavior are known upfront, metrics such as entropy are an effective implementation of this paradigm. However, in an unsupervised setting when classes of behavior

are not known, it is much more challenging to adhere to the clustering evaluation paradigm. Both accuracy and complexity metrics presented earlier quantify intra-cluster similarity. However, there does not exist a standard approach to assess the minimization of the inter-cluster similarity in trace clustering. It is out of the scope of this study to put forward such an evaluation framework. Note again that the proposed evaluation approach is similar to [15, 16].

4.5 Experimental Evaluation

In this section, the techniques proposed in this study are evaluated in two different settings. First of all, a controlled environment is devised in which the parameter configurations of the ActiTraC-algorithms are analyzed. Furthermore, this supervised experiment is used to assess the capabilities of ActiTraC and existing techniques of identifying ex ante defined classes of behavior. The remainder of this section describes a scalability assessment of ActiTraC and a benchmarking study based on 4 real-life event logs. Note that the trace clustering techniques presented in this chapter are implemented as the *ActiTraC plugin* in ProM 6[1], an academic process mining framework which allows the use and development of a large variety of techniques for analyzing event logs. An illustration of the parameter configuration and the output view of the ActiTraC plugin is shown in Figure 4.2.

4.5.1 Controlled Environment Evaluation

Because there does not exist a comprehensive benchmarking environment for trace clustering techniques in the process mining literature, it is opted to design a dedicated controlled evaluation environment. Such an experimental evaluation allows for assessing whether the trace clustering techniques are capable of identifying ex ante defined classes of behavior.

Design

A first essential design choice consists of the definition of different classes of behavior. Therefore, traces have to be generated so as to adhere to a certain class of behavior as determined by a *control-flow based* domain criterion. Such criteria can be very diverse, e.g. presence or absence of activities, presence or absence of combinations of activities, looping behavior, etc. In the context of this assessment, it was decided to opt for a straightforward criterion, namely the presence or absence of a single activity. In this way, the Petri net in Figure 4.3 was designed to generate different event logs with 3 classes of behavior: traces where activity

[1] http://www.promtools.org/prom6/

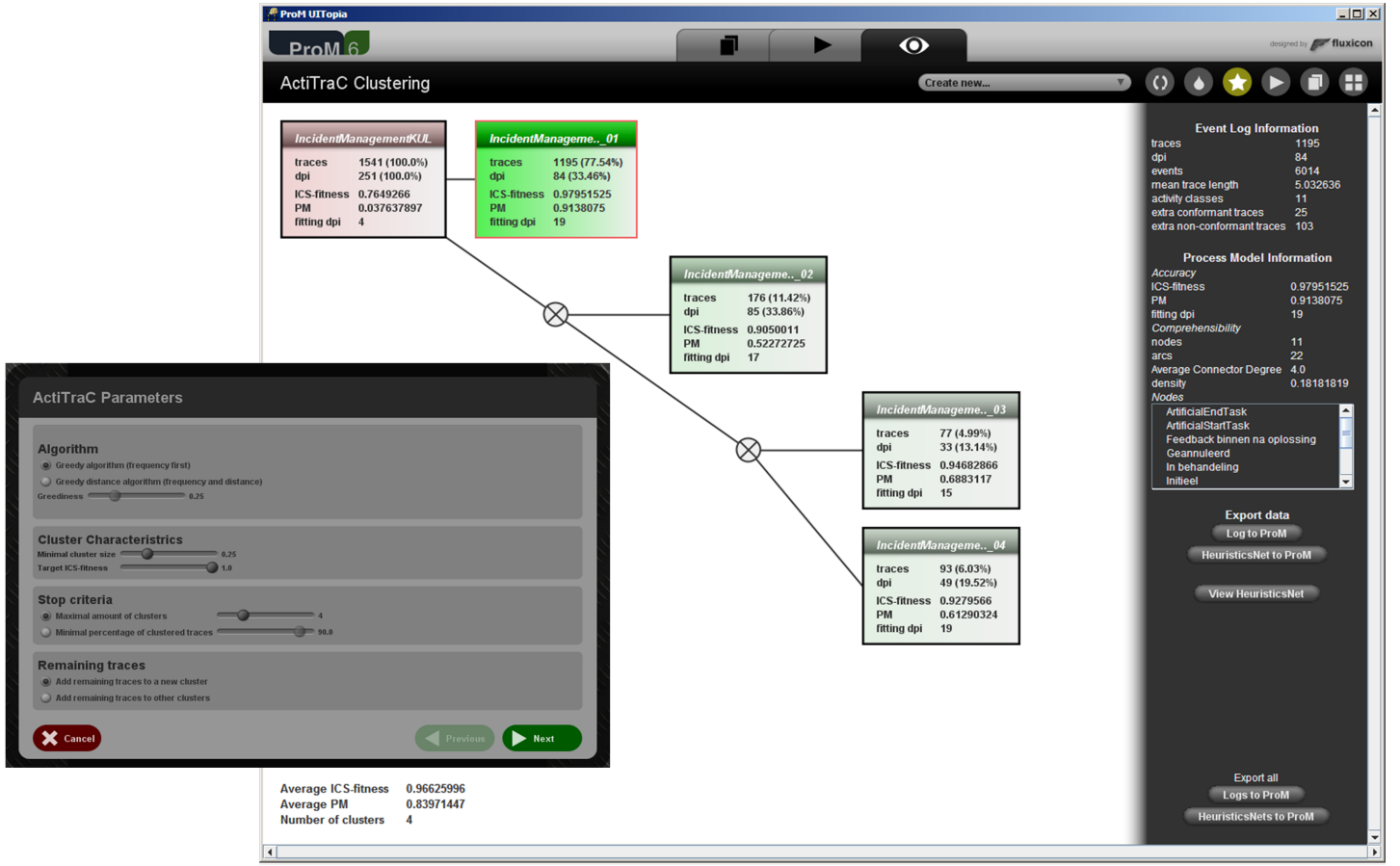


Figure 4.2: Parameter configuration (left) and output view (right) of the ActiTraC plugin in ProM 6

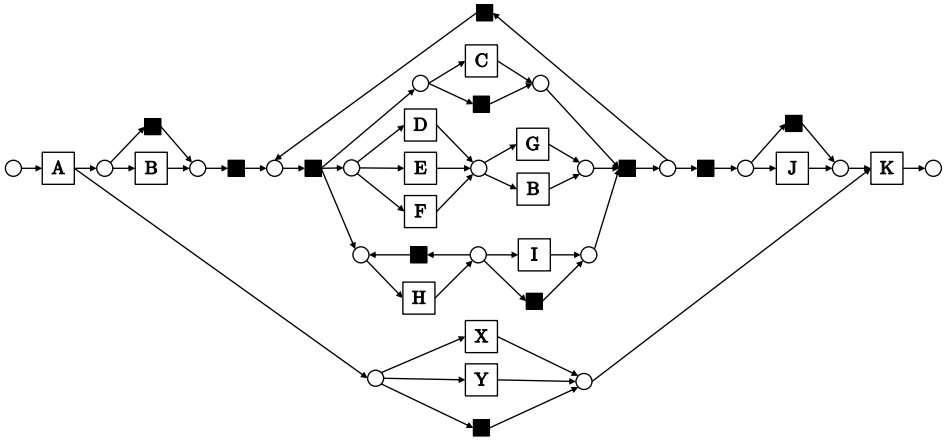


Figure 4.3: Petri net used for generating different classes of behavior according to the presence/absence of activities X and Y

X is present, traces where activity Y is present and traces where neither activity X nor activity Y is present. Note that it was opted to include plenty of parallel and looping behavior in order to mirror the complexity of a real-life event log.

Using this Petri net, three different event logs were generated randomly using CPN Tools[2], containing 2.000 cases of each behavioral class (6.000 cases in total) and with each behavioral class accounting for 200 distinct process instances. The three event logs differ in terms of the distribution of distinct process instances. The exponential (linear) log presents a an exponentially (linearly) decreasing distribution in terms the frequency of the distinct process instances, while for the uniform log each distinct process instance accounts for a frequency of 10.

Results

Within this controlled environment, different parameter settings for the ActiTraC algorithm are evaluated by verifying how well ActiTraC is able to identify the different classes of behavior as well as by investigating the accuracy improvements realized by clustering traces. Moreover, the performance of ActiTraC in this environment is compared to 7 existing trace clustering techniques.

Parameter settings. For ActiTraC, three parameters are considered essential: *window size*, *minimum cluster size* and *target fitness*. As such, different values for these parameters were evaluated according to Cluster Set Entropy and Weighted Average F1-scores for the 3 artificial event logs, as shown in Figure 4.4. When taking into account the ability of ActiTraC to distinguish between the induced

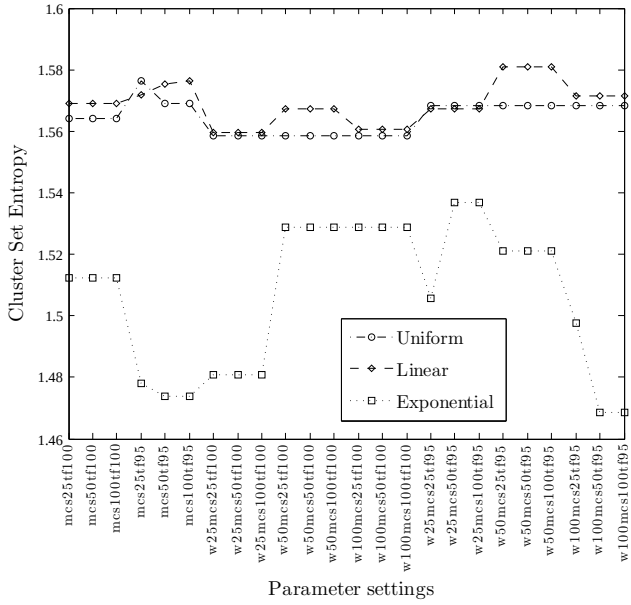
[2]http://www.cpntools.org

clusters, the performance is rather weak. Especially for the uniform and linear event log, the Cluster Set Entropy is very close to the maximal entropy of 1.585. On the other hand, the exponential case provides us with some more insights into the behavior of the algorithm. For instance, for the pure frequency-based selective sampling (the first six settings on the x-axis), a decrease in terms of the target fitness has a positive effect on the entropy. For the MRA-based sampling procedure, the effects of window size and minimum cluster size depend on the target fitness. In case of a target fitness of 1, a decreased window size tends to improve Cluster Set Entropy, while for a decreased target fitness, an increase of the window size tends to affect Cluster Set Entropy positively.

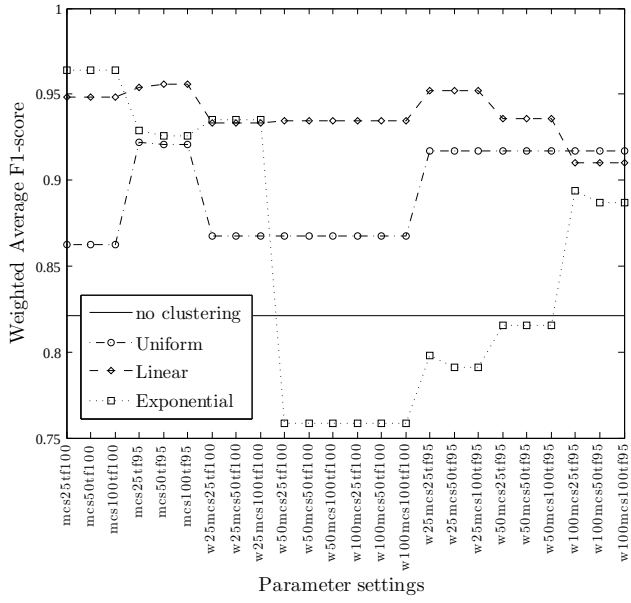
Next to entropy, the accuracy results realized with different values of the ActiTraC parameters are detailed in Figure 4.4b. For the more challenging uniform and linear logs, it holds that a decrease in target fitness has a positive effect on the F1-scores. This is mostly due to the difficulty for ActiTraC to join high numbers of traces in one single process model with perfect fitness. This is because the clusters formed by ActiTraC with the target fitness set to 1 are founded on a limited set of traces which causes the underlying models to be inaccurate for the entire set of traces in the cluster. When decreasing the target fitness, clusters are created based on a higher number of traces, which has a positive effect on the F1-score. This shows that event logs consisting of more challenging control-flow behavior for process discovery techniques might benefit from decreasing the target fitness. Note that the solid, non-marked, horizontal line denotes the average F1-score over the three event logs for the baseline scenario where no clustering is performed.

In the case of a more or less exponential distribution of trace frequencies, the effects of decreasing the target fitness are reversed. For both selective sampling strategies, the best results can be obtained by setting the target fitness to 1. This is coherent with the design of ActiTraC which employs trace frequencies as a steering information source for selecting traces and optimizing the underlying process models. Finally, note that the effects of the minimum cluster size are limited because of the difficulty of the artificially generated event logs. Due to the pure randomness of trace generation, the ActiTraC algorithm is unable to find a sufficient amount of traces that can be clustered with the selected target fitness. In other situations, enlarging the minimum cluster size could be useful to increase the total amount of traces on which the underlying process models are learned from.

Benchmark performance. Next to the evaluation of different parameter settings for ActiTraC, the closed environment is employed to benchmark the performance with other clustering techniques. Based on the previous analysis, standard parameter settings for both ActiTraC$^{\text{freq}}$ and ActiTraC$^{\text{MRA}}$ are defined as follows: target fitness is set to 1, minimum cluster size is set to 25% and for ActiTraC$^{\text{MRA}}$



(a) P/T-CD$_A$ (lower is better)



(b) F1$_{W.A.}$ (higher is better)

Figure 4.4: Average Entropy and Weighted Average F1-scores for different parameter settings of ActiTraC and for 3 distinct event logs with varying frequency distributions of the dpi's

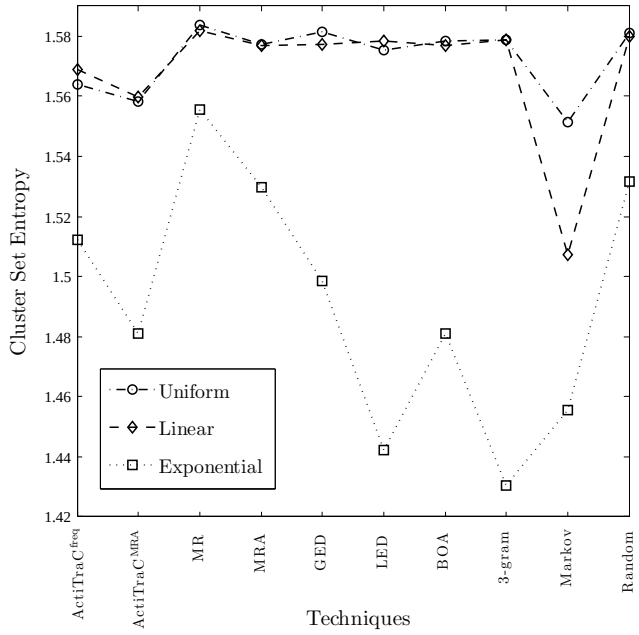
a window size of 25% is set. Despite the fact that in some situations, it might be useful to change these values, standard parameters are used in the remaining analyses. Note that these settings are not *optimal* for each event log individually. The development of a parameter optimization strategy that can deal with possibly large computation times required for clustering and evaluation metric calculation is out of the scope of this study.

ActiTraC is compared to 7 clustering techniques: MR and MRA [16], GED and LED [15], bag of activities (BOA) and 3-gram [120], and the Markov chain clustering technique from [55]. It should be noted that agglomerative hierarchical clustering with minimum variance is used as the underlying clustering technique for BOA, 3-gram, GED, LED, MR and MRA. Despite the fact that other underlying clustering techniques might increase performance, this is not considered in this study. For BOA, 3-gram, MR and MRA, the best performing parameter settings are employed for each evaluation criterion and for each event log individually by varying the vector representation (i.e. binary or numeric) and the underlying distance metric (i.e. F-score similarity or Euclidean distance). The application of feature filtering is not considered. Finally, the average results of a 5-fold repetition of randomly clustering traces into different groups is also added (Random).

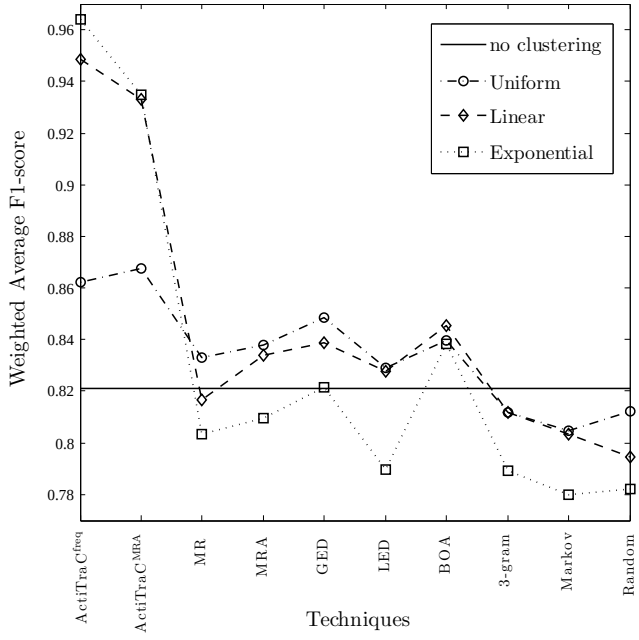
The results of the comparative analysis are depicted in Figure 4.5. Taking into account the Cluster Set Entropy, it is concluded that the benchmark techniques tend to face the same difficulties as ActiTraC with respect to distinguishing the different classes of behavior. Only the Markov clustering technique outperforms the ActiTraC algorithms on every artificial data set. However, with a best Cluster Set Entropy of only 1.44 (3-gram), the main conclusion of this experiment is that all assessed clustering techniques are incapable of adequately detecting the predetermined classes. It is pointed out that clustering techniques might divide traces into different clusters based on other domain-specific criteria. Accordingly, we cannot draw general conclusions with respect to the domain relevancy of different clustering techniques.

Figure 4.5b presents the accuracy results in terms of the Weighted Average F1-score. Note that this metric is calculated by first applying HeuristicsMiner (with standard parameter settings) to the clusters and then translating the resulting model into a Petri net by making use of a translation plugin in ProM. Next, recall and precision metrics are calculated and combined into an individual F1-score for each cluster. As detailed in Section 4.4.2, these F1-scores can be weighted so as to compute the Weighted Average F1-score for each of the solutions of the clustering techniques. Note that for ActiTraC, the process models underlying each of the clusters are calculated within the clustering algorithm.

In the figure, the effects of bridging the gap between clustering bias and evaluation bias are clearly observable. Even for the most challenging setting for ActiTraC, i.e. the uniform event log where there is no frequency difference between



(a) Average Entropy (lower is better)



(b) $F1_{W.A.}$ (higher is better)

Figure 4.5: Comparison of ActiTraC with other trace clustering techniques according to Average Entropy and Weighted Average F1-scores

the traces and thus the selection is essentially random for ActiTraC$^{\text{freq}}$ and purely MRA-driven for ActiTraC$^{\text{MRA}}$, both techniques outperform every other available clustering technique, even with suboptimal parameter settings. As soon as trace frequencies start to differ among traces, this effect is magnified.

4.5.2 Scalability

Before further assessing the capabilities of ActiTraC on real-life event logs, its scalability is examined. Note that in Section 4.3.5, it was already pointed out that some essential design choices of the algorithm, namely its underlying process discovery technique and its underlying fitness score are mainly chosen for scalability reasons. Regarding complexity, the core of the ActiTraC algorithm scales linearly in terms of the number of *dpi*'s as well as in terms of the number of clusters. However, the overall computational complexity of our algorithm depends largely on the underlying techniques, i.e. the process discovery technique, the fitness calculation and the trace selection procedure. As such, considering the current choices, ActiTraC scales linearly in terms of the number of clusters and quadratically in terms of *dpi*'s and in terms of activity types (at).

The computational complexity of ActiTraC

This paragraph details the computational complexity analysis of our algorithm in term of the big O notation Hereto, we make use of the following notation: nb_{clus} denotes the number of clusters, dpi denotes the number of distinct process instances in the event log, $SSST$ denotes the selective sampling strategy, PDT denotes the process discovery technique and FC denotes the fitness calculation. As such, the computational complexity of the different subphases of ActiTraC can be described as follows:

- Phase 1 - Selection:

$$O\left[nb_{clus} \times dpi \times \left(f\left(SSST\right) + f\left(PDT\right) + f\left(FC\right)\right)\right]$$

- Phase 2 - Look ahead:

$$O\left[dpi \times \left(f\left(FC\right)\right)\right]$$

- Phase 3 - Residual trace resolution:

$$O\left[nb_{clus} \times dpi \times \left(f\left(FC\right)\right)\right]$$

Aggregation of these complexities results in:

$$O[nb_{clus} \times (dpi \times (f(SSST) + f(PDT) + f(FC)) + dpi \times (f(FC)) + nb_{clus} \times dpi \times (f(FC)))]$$

$$O[nbclus \times (dpi \times (f(SSST) + f(PDT) + f(FC)) + dpi \times (f(FC)) + nbclus \times dpi \times (f(FC)))]$$

Simplifying results in:

$$O[nb_{clus} \times dpi \times (f(SSST) + f(PDT) + f(FC))]$$

As one can see, our algorithm scales linearly in terms of number clusters and linearly in terms of *dpi*. Furthermore, computational complexity of ActiTraC depends on the computational complexity of either the process discovery technique, the fitness calculation or the selective sampling procedure, depending on the specific scalability of each of these functions.

For instance, instantiation to the current implementation of ActiTraC gives:

$$O[nb_{clus} \times dpi \times (f(\textit{MRA-based Euclidean distance}) + f(HM) + f(\textit{ICS-fitness}))]$$

Assuming that the scalability of HeuristicsMiner is worse than the selective sampling method and the fitness calculation and assuming that HeuristicsMiner scales linearly in terms of *dpi*'s and quadratically in terms of activity types (at), the time complexity of ActiTraC is as follows:

$$O\left[nb_{clus} \times dpi^2 \times at^2\right]$$

As such, it can be concluded that the ActiTraC algorithm scales linearly in terms of the number of clusters and linearly in terms of *dpi*, but with the inclusion of HeuristicsMiner as the underlying discovery technique, ActiTraC scales quadratically in terms of *dpi*'s and in terms of *at*'s.

An experimental analysis of ActiTraC's scalability

Next to a theoretical analysis of the computational complexity, an experimental analysis was performed as well. Figure 4.6 details a scalability analysis of ActiTraC for its worst case scenario. This scenario is determined by setting the

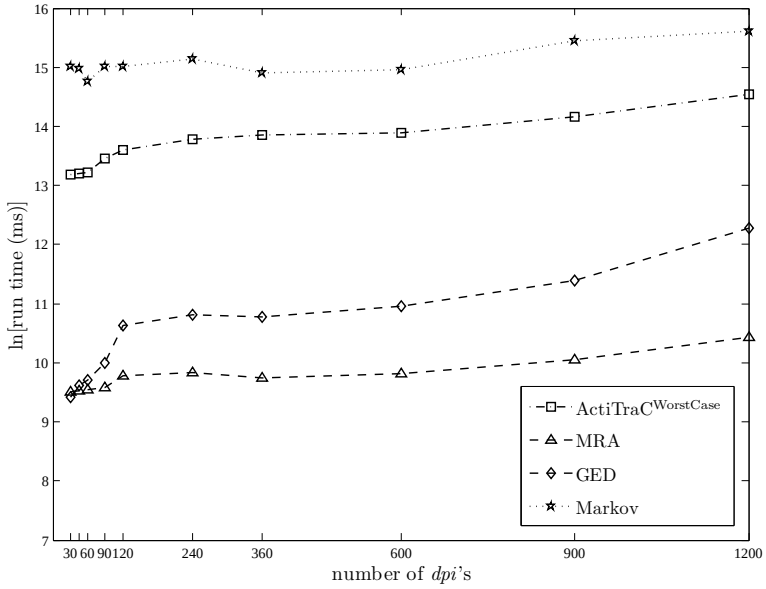
window size to 1 and the target fitness to $-\infty$. In this case, none of the implemented greedy heuristics are used so that a maximal amount of selective sampling procedures needs to be executed. The scalability assessment consists of the creation of 100 artificial event logs with varying number of activity types and varying number of distinct process instances. Note that with these settings, the effect of the underlying distribution of trace frequencies is canceled out. By focusing on distinct process instances and activity types, the two main complexity criteria for ActiTraC are taken into account.

Figure 4.6a depicts the run time results of 4 clustering algorithms varying the number of distinct process instances. Note that the plot uses a logarithmic scale. Each marker in this figure represents the average calculated over 10 different event logs with varying number of activity types. Figure 4.6b mirrors this setting for the number of activity types averaging out over 10 logs with varying number of distinct process instances. The results clearly show an increased complexity for ActiTraC as compared to contemporary clustering algorithms. Note that MRA is shown as a representative for both GED as well as for all other vector space approaches because of very similar performance results. In conclusion, ActiTraC in its worst case scenario requires up to 50 times as much computational resources in comparison to vector space approaches. Nonetheless, ActiTraC does outperform the Markovian model-based clustering technique, which is the most similar of all other clustering techniques in terms of the clustering procedure. Note that the scalability assessment was performed on a standard stand-alone PC with a 2.83Ghz quad-core CPU and 3GB of RAM.

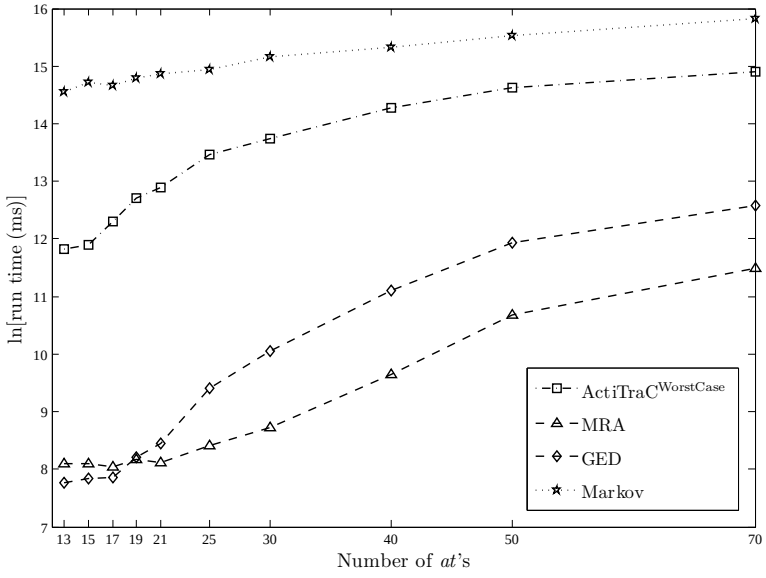
4.5.3 Real-Life Event Logs

The final part of the experimental evaluation of ActiTraC consists of its application to 4 real-life event logs. Table 4.1 shows some basic statistics and a description of the event logs used. It is important to note that the event logs originate from flexible environments exhibiting a large variety of process behavior, as illustrated with the metrics in the table. Because in real-life settings, ex ante classes of behavior are unknown, the use of entropy-based evaluation is infeasible. Therefore, the evaluation is centered on two dimensions, i.e. accuracy in terms of the Weighted Average F1-score and complexity in terms of the Average Place/Transition Connection Degree.

The experiments are set up in a similar way as the comparative assessment in the controlled environment. For the 4 real-life event logs, 10 different trace clustering approaches were applied varying the maximal amount of clusters from 3 to 5. For both ActiTraC$^{\text{freq}}$ and ActiTraC$^{\text{MRA}}$ the standard parameters are used while for the other techniques the best performing parameter settings are employed for each data set and each cluster size individually. This setup emphasizes ActiTraC's



(a) Distinct Process Instances



(b) Activity Types

Figure 4.6: Scalability comparison of ActiTraC for its worst case scenario by varying the number of distinct process instances and the number of activity types

Label	Event log properties				Organization	Process description
	# pi	# ev	# at	# dpi		
KIM	24.770	124.217	18	1.174	KU Leuven	Helpdesk process of the ICT service
MCRM	956	11.218	22	212	Manufacturing company	CRM process
TSL	17.812	83.286	42	1.908	Telecom company	Second-line CRM process
ICP	12.391	65.653	70	1.411	Insurance company	Incoming document handling

Table 4.1: Description of the real-life event logs with following characteristics: the number of process instances (# pi), number of events (# ev), the number of activity types (# at), and the number of distinct process instances (# dpi)

robustness.

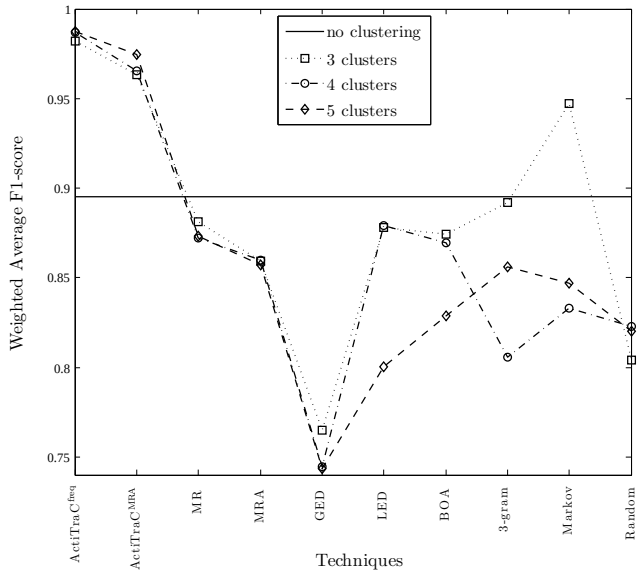
Accuracy results

The accuracy results are presented in Figure 4.7. The solid, non-marked, horizontal line denotes the base case with no trace clustering where HeuristicsMiner is applied to the whole event log. In general, it is concluded that the ActiTraC algorithms significantly outperform the reference clustering algorithms with respect to process model accuracy. We observe consistent performance improvements considering both the different event logs and the varying number of clusters.

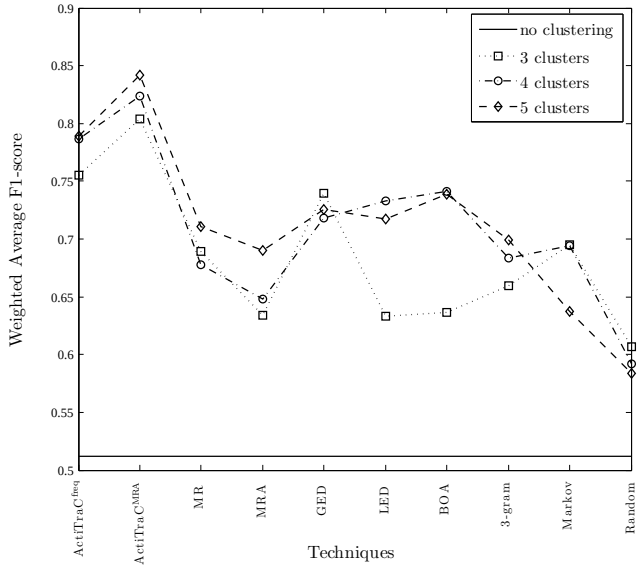
Remark also that for many of the existing trace clustering techniques, the Weighted Average F1-score is rather poor. As such, for event logs KIM and TSL, almost all other techniques perform worse than the case of not clustering at all. Also, in many situations, these techniques cannot significantly outperform the random clustering of traces. This is clear evidence for the initial observation that there is a strong divergence between the clustering bias and the evaluation bias for these techniques. As for ActiTraC, it is concluded that in most cases, the pure frequency-based selective sampling strategy will slightly outperform the MRA-based selective sampling strategy from an accuracy perspective. However, the smaller MCRM log shows that it can be advantageous to force the selection of more similar traces first.

Complexity results

Our evaluation approach also consists of an assessment of the comprehensibility of the discovered process models. As explained in Section 4.4, the Average Place/Transition Connection Degree is employed as a quantification of process model complexity. The results of this complexity analysis are presented in Figure 4.8. For these plots, the lower the value, the better the performance because a

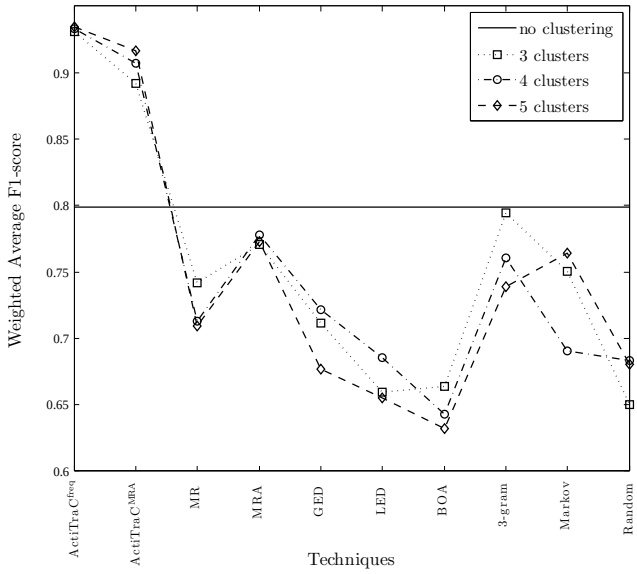


(a) KIM

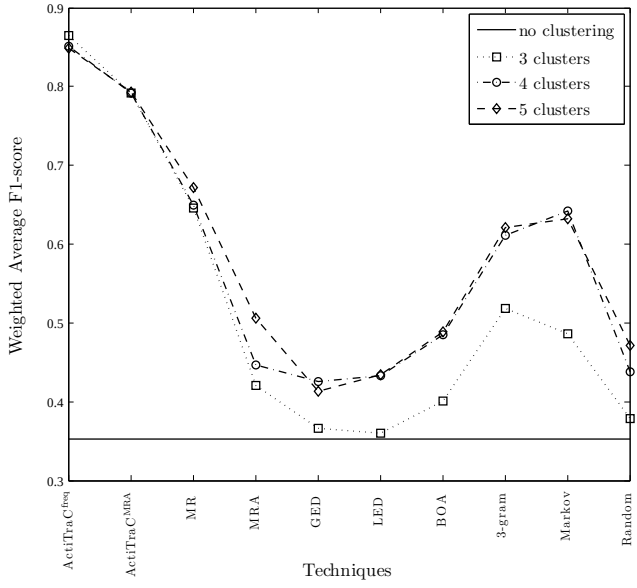


(b) MCRM

Figure 4.7: Weighted Average F1-scores for the 4 real-life event logs. Note that the ActiTraC algorithms clearly outperform the other clustering techniques in all the benchmarks (higher is better)



(c) TSL



(d) ICP

Figure 4.7: Weighted Average F1-scores for the 4 real-life event logs. Note that the ActiTraC algorithms clearly outperform the other clustering techniques in all the benchmarks (higher is better)

Table 4.2: Average run times with 95% confidence interval for the real-life event logs with 4 clusters

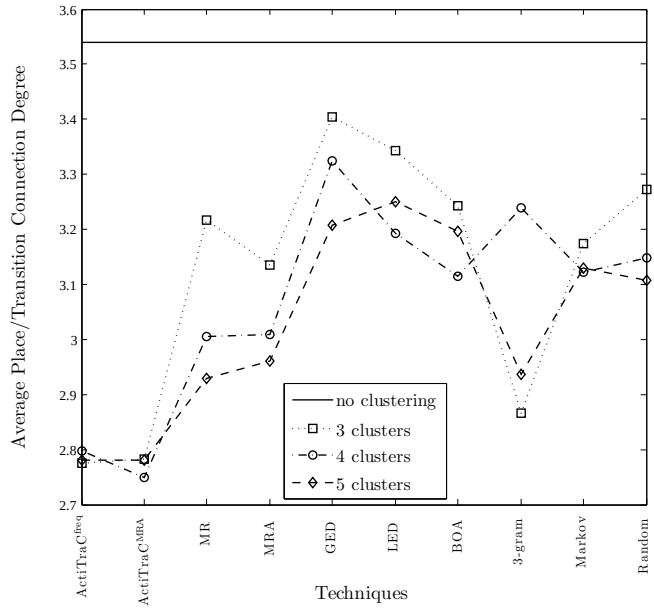
Technique	KIM	MCRM	TSL	ICP
ActiTraC^{freq}	00:04:39,912 ± 6,568s	00:01:01,739 ± 2,163s	00:05:02,383 ± 15,694s	00:05:48,682 ± 12,207s
ActiTraC^{MRA}	00:02:58,496 ± 1,306s	00:00:13,328 ± 0,052s	00:06:11,680 ± 1,671s	00:10:18,691 ± 5,099s
ActiTraC^{WorstCase}	01:27:58,290 ± 9,603s	00:01:18,764 ± 0,041s	02:04:35,266 ± 48,694s	01:02:36,146 ± 8,389s
MR	00:01:16,214 ± 0,525s	00:00:01,907 ± 0,005s	00:03:59,391 ± 1,070s	00:01:12,726 ± 0,337s
MRA	00:00:33,384 ± 0,376s	00:00:01,778 ± 0,014s	00:01:59,582 ± 0,501s	00:00:56,089 ± 0,328s
GED	00:00:41,952 ± 0,068s	00:00:03,578 ± 0,046s	00:01:10,673 ± 0,467s	00:00:31,556 ± 0,114s
LED	00:00:07,600 ± 0,058s	00:00:00,391 ± 0,001s	00:00:22,171 ± 1,043s	00:00:09,297 ± 0,087s
BOA	00:00:06,472 ± 0,223s	00:00:00,244 ± 0,010s	00:00:22,324 ± 0,184s	00:00:11,309 ± 0,270s
3-gram	00:00:21,862 ± 0,070s	00:00:00,515 ± 0,000s	00:02:54,665 ± 0,847s	00:01:33,392 ± 0,384s
Markov	08:22:40,000 ± 9334,281s	00:00:51,000 ± 13,917s	04:35:14,000 ± 4042,004s	01:42:56,000 ± 756,172s

decrease in Average Place/Transition Connection Degree signifies a less complex process model.

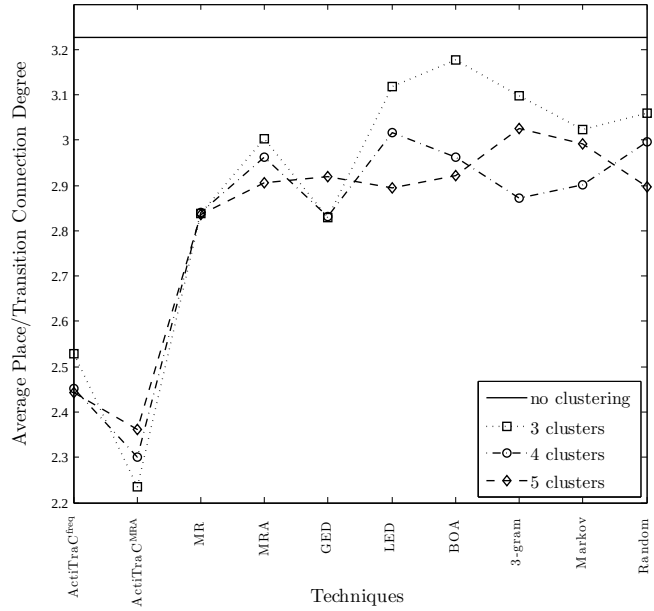
From the figure, it can be seen that also from a complexity viewpoint, the results clearly favor the ActiTraC algorithms. For the 4 real-life logs, the additional decrease in complexity of the Petri net models as compared to the other clustering techniques is significant. With respect to the difference between the ActiTraC techniques themselves, there is a slight indication that MRA-based sampling has the potential to create clusters for which the underlying process models are less complex. Only for the ICP event log, this pattern cannot be observed.

Scalability

Because the computational complexity of ActiTraC is a crucial factor determining its applicability in practice, Table 4.2 provides an overview of the run times of the different techniques for the creation of 4 clusters for each of the real-life event logs. From these results, it can be concluded that the greedy heuristics underlying the ActiTraC algorithm are effective in terms of restricting the computational requirements.

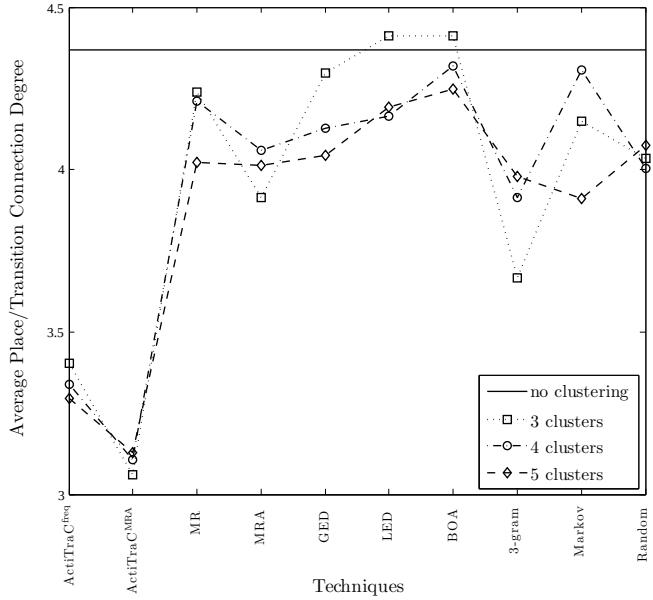


(a) KIM

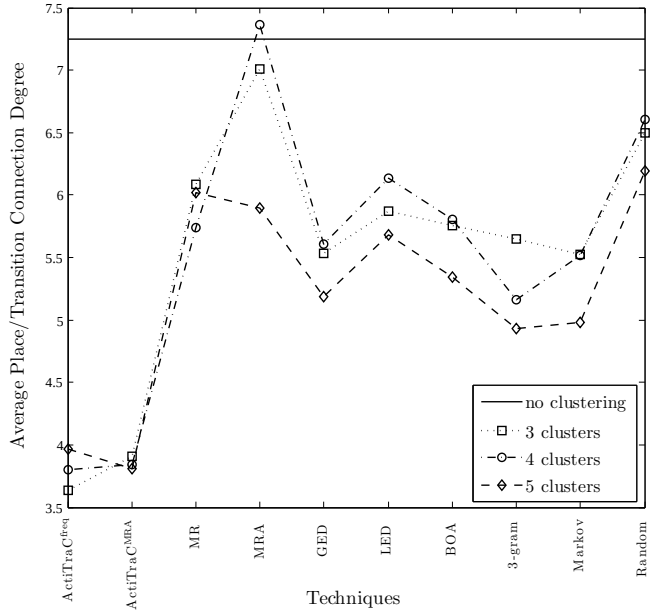


(b) MCRM

Figure 4.8: Average Place/Transition Connection Degree for the 4 real-life event logs. Note that the ActiTraC algorithms clearly outperform the other clustering techniques in all the benchmarks (lower is better)



(c) TSL



(d) ICP

Figure 4.8: Average Place/Transition Connection Degree for the 4 real-life event logs. Note that the ActiTraC algorithms clearly outperform the other clustering techniques in all the benchmarks (lower is better)

4.6 Discussion

To conclude, the presented trace clustering technique achieves its goal to improve currently available trace clustering techniques with respect to evaluating from a process discovery perspective. Our approach changes the objective of traditional trace clustering to find an optimal distribution in terms of grouping similar traces together to the goal of solving the problem of finding an optimal distribution of traces so as to maximize the combined accuracy of the underlying process models. We think that in many situations, the latter objective is useful because the challenge of finding a criterion that separates process instances according to their intrinsic similarity is often made difficult due to the data explosion problem. This problem was not only demonstrated with the experiment in the supervised environment but is also recognized in [58] where outlier detection is proposed as a solution for the problem.

Nevertheless, it is acknowledged that grouping homogeneous traces together is useful. Therefore, ActiTraC has been provided with the flexibility to take into account any kind of distance function between instances. With respect to addressing the challenge of combined accuracy and homogeneity optimization, we are convinced that a more comprehensive benchmarking environment for trace clustering techniques should be devised. Currently, such a framework is not available in the process mining literature.

A final element of discussion is cluster characterization. For trace clustering to be a genuine approach towards solving the complexity problem arising from highly unstructured event logs, process analysts should be able to delineate the discovered clusters to give an interpretation to the solution. Accordingly, in the next section, an analysis is provided on how trace clusters can be characterized. Also, a rule learning algorithm is shown to be useful to ex post characterize the differences between groups of process instances.

4.7 Cluster Characterization

Qualifying the differences between the discovered process models in the clustering solution plays a vital role for interpretability and thus usefulness of trace clustering. In this section, three possible methods are identified that are capable of typifying differences between sub-event logs in a clustering solution.

4.7.1 Visual Analysis of the Process Models

A straightforward solution for cluster characterization is the visual analysis of the resulting process models. By a visual inspection, differences between process models and the according trace clusters can be described. The implementation of

ActiTraC in ProM 6 allows to visualize and store the process models. However, although the visual inspection of the discovered process models is an intuitive solution, different drawbacks exist. Foremost, even for two not overly complex process models, it is all but easy to identify the key differences. As these dissimilarities might reside in small details, it is doubtful whether non-experts are able to qualify process model differences based on a visual analysis.

4.7.2 Quantifying Similarity of Business Process Models

A second solution for cluster characterization is the use of similarity metrics between business process models. Dijkman et al. [47] and Alves de Medeiros et al. [8] propose different metrics to quantify process model differences. Dijkman et al. provide three different similarity metrics: node matching similarity, structural similarity and behavioral similarity. In contrast, Alves de Medeiros et al. define similarity metrics by comparing two process models based on some typical behaviors. However, as with other studies on quantifying process model similarity, the main issue with these metrics is that they do not provide any details about the root causes of process dissimilarity. Accordingly, it is difficult to apply these metrics for cluster characterization.

4.7.3 A Rule Set Approach

Since there does not exist a clear strategy on how to compare process models and thus interpreting the result of a trace clustering solution, a distinctive approach is proposed. We therefore employ reverse engineering of a typical trace clustering approach as employed by Greco et al. [66] and Song et al. [120]. Their clustering technique relies on the construction of a vector space model based on activities and activity transitions. By applying this same vector space model but with the cluster as a dependent variable, it is possible to apply a rule learner such as RIPPER [30].

By applying RIPPER, a comprehensible set of rules can be learned based on activities and activity transitions that explains to some extent how trace clusters and thus the respective process models differ. Since the observations are vectors corresponding to the distinct process instances, there is an important problem of class imbalance. This is because the different trace clusters will often differ in terms of the number of distinct process instances. In order to resolve this issue, we applied an oversampling method that involves the replication of observations in order to balance the number of observations over a set of trace clusters. In this way, majority class predictions are avoided so that more useful results are derived from RIPPER.

It is recognized that other researchers have investigated the use of rules within

the domain of process mining. For instance, Rozinat et al. [113] show how decision trees can be used for discovering rules that discriminate between possible actions at the decision points in a certain process model. Furthermore, Liu [86] describe a case study where rule learning techniques are employed to find patterns that discriminate between succeeded and failed process instances. This approach is implemented in ProM as the Signature Discovery plugin.

Results. In Figure 4.9, the results are shown of the outlined approach being applied to event log KIM with the maximal amount of clusters set to four. The plot shows the percentage of correctly classified distinct process instances given a certain number of rules for different ActiTraC variants, standard MRA-based clustering and clustering based on activities and activity transitions. The latter clustering algorithm is denoted *BOA/AT*. Note that in this case, four variants of ActiTraC are inspected by also varying the option to distribute the remaining traces over existing clusters. Accordingly ActiTraC$^{\text{freq}}$ denotes the frequency-based algorithm without redistribution of remaining traces and ActiTraC$^{\text{freq+}}$ denotes the case with redistribution. In turn ActiTraC$^{\text{MRA}}$ denotes MRA-based selective sampling without redistribution while ActiTraC$^{\text{MRA+}}$ denotes MRA-based selective sampling with redistribution.

In general, it can be concluded that the approach is able to explain trace clusters with relatively good accuracy. Note that the accuracy is calculated based on the non-oversampled data set. In Figure 4.10, an example RIPPER rule set is given with a single character representing an activity and double character combination representing an activity transition. Furthermore, Figure 4.10 also shows some interesting results in terms of comparing the different ActiTraC variants. For instance, it can be concluded that the redistribution of remaining traces noticeably complicates the interpretation of the trace clusters by RIPPER rules. A more counterintuitive result is that the MRA-variants have a tendency to create clusters that are slightly more difficult to interpret given the choice to only make use of activities and activity transitions. Probably, this is due to the fact that MRA employs more complex contexts of execution to create the set of attributes that are used for calculating the distances between traces. This is confirmed by the fact that the original MRA-based agglomerative hierarchical clustering technique underperforms as well. Note that this observation contrasts with the complexity results in Section 4.5.3 where the Average Place/Transition Connection Degree was used to assess the complexity of the underlying process models.

As a final remark, it is pointed out that the outlined cluster characterization approach presents some shortcomings as well. Firstly, by making use of rule sets, it might not always be possible to characterize trace clusters, especially in case of large and complex event logs. Furthermore, the proposed approach only takes into account presence of a single activity and succession of two activities as at-

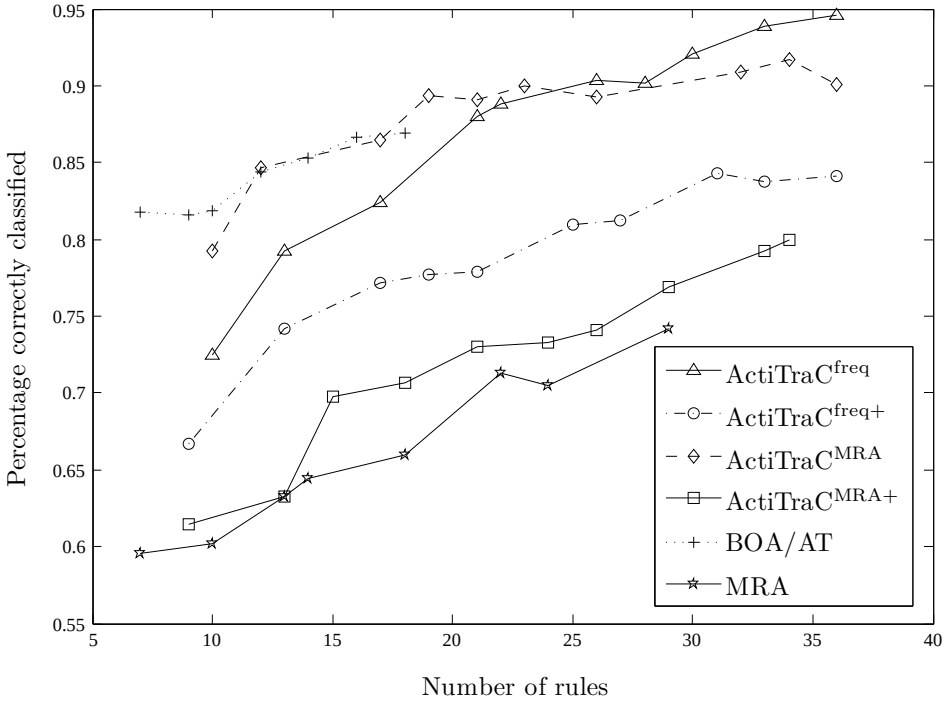


Figure 4.9: Rule set accuracy according to the number of rules, as applied to event log a

$$
\begin{aligned}
(nm = 1) \wedge (k = 0) \wedge (mb = 1) \wedge (o = 0) \wedge (mm = 0) \wedge (dm = 1) &\Rightarrow cluster = 2\\
(nm = 1) \wedge (k = 0) \wedge (mb = 1) \wedge (op = 0) \wedge (no = 0) \wedge (mn = 1) \wedge (c = 0) \wedge (mm = 0) \wedge (nn = 0) \wedge (mp = 0) &\Rightarrow cluster = 2\\
(qg = 1) \wedge (m = 0) \wedge (p = 0) &\Rightarrow cluster = 2\\
(cj = 1) \wedge (dc = 1) &\Rightarrow cluster = 2\\
(ng = 1) \wedge (nm = 1) \wedge (gn = 0) &\Rightarrow cluster = 2\\
(p = 0) \wedge (n = 0) \wedge (q = 0) \wedge (dl = 0) \wedge (n = 0) \wedge (j = 0) &\Rightarrow cluster = 0\\
(om = 1) \wedge (n = 0) \wedge (p = 0) \wedge (mb = 1) \wedge (k = 0) \wedge (mo = 0) &\Rightarrow cluster = 0\\
(k = 0) \wedge (n = 0) \wedge (c = 0) \wedge (po = 0) \wedge (b = 1) &\Rightarrow cluster = 1\\
(k = 0) \wedge (lr = 1) \wedge (p = 0) \wedge (m = 0) &\Rightarrow cluster = 1\\
(nb = 1) \wedge (k = 0) \wedge (om = 0) \wedge (p = 0) \wedge (nm = 0) \wedge (no = 0) \wedge (c = 0) &\Rightarrow cluster = 1\\
(qm = 1) \wedge (n = 0) \wedge (qk = 0) \wedge (p = 0) &\Rightarrow cluster = 1\\
otherwise &\Rightarrow cluster = 3
\end{aligned}
$$

Figure 4.10: An example rule set with 85% accuracy for the ActiTraC$^{\text{MRA}}$ algorithm as applied to event log KIM

tributes for rule learning. It can be expected that the inclusion of other attributes might further improve this methodology. For instance, the use the entire range of DECLARE templates (see [87]) has the potential to boost the performance of rule learning for cluster characterization.

4.8 Conclusion

In this chapter, we have proposed a new approach to trace clustering. Trace clustering is one possible strategy to resolve the problem that currently available process discovery algorithms are unable to discover accurate and comprehensible process models out of event logs stemming from highly flexible environments. The technique, called ActiTraC, has its foundations in the observation that currently available trace clustering algorithms suffer from a severe divergence between the clustering bias and the evaluation bias. This problem is resolved with an active learning inspired approach that centers on optimizing the combined process model accuracy. In the experimental evaluation, both artificial as well as real-life event logs were used to show that ActiTraC, even without optimization of its parameters, significantly improves the accuracy and complexity of the process models underlying the discovered trace clusters as compared to existing trace clustering techniques.

Chapter 5

Case Studies

In this chapter, three real-life case studies are presented that further validate the work in the previous chapters. The first study is situated in the financial services industry with the primary objective of this study being a demonstration of the applicability and usefulness of process mining in a large scale service company. A second case study presents the analysis of an event log containing clinical pathway data from a healthcare process. Herewith, it is shown how the analysis of low-granular event data can be embarked on. Furthermore, the use of network visualizations is shown to be an interesting approach for knowledge discovery and knowledge representation. Finally, the third case presents an application study of ActiTraC in which trace clustering is combined with text mining and decision tree learning so as to characterize deviating process behavior. For this last case study, data from the ICTS helpdesk from the KU Leuven has been used.

5.1 Process Mining for the Multi-Faceted Analysis of Business Processes: A Case Study in a Financial Services Organization

These days it is impossible for an organization to operate without some sort of enterprise information system. During the last decades, information systems (IS) have transformed from simple systems with limited functionality to complex, integrated architectures. As a result, it becomes harder to understand and monitor how these systems impact the execution of every-day processes in organizations. Process Mining [135] offers a solution based on the extraction, analysis, diagnosis and visualization of the data recorded by an IS during process execution. Although in the past, major contributions to the process mining literature were predominantly technical in nature, techniques have proved their usefulness in practice as well. Nevertheless, application-oriented studies have only received modest attention and therefore, this study demonstrates the benefits and challenges of applying process mining techniques in practice by a multi-faceted analysis of business processes within the back office of a Belgian insurance company.

Process Mining goes beyond the capabilities of traditional business intelligence tools [64] with respect to process analysis. Accordingly, it can be considered as a proficient means for helping organizations understanding their actual way of working and thereby serving as a foundation for process improvement. This is mainly due to the fact that the cornerstone of Process Mining is real data that comprises how business operations are actually carried out in an organization. This is significantly different from other approaches to process improvement, for instance relying on interviews with key stakeholders.

Based on existing literature and our own experiences, a methodology framework is described, which structures the process mining study in a financial services organization. This framework is similar to earlier works [19, 105], however it puts an emphasis on data extraction and exploration as well as on the multi-faceted nature of analyzing process execution data. Furthermore, this study clarifies benefits as well as challenges of conducting a real-life process mining study. For instance, the study points out the importance of intensive two-way communication between process analysts and organization's experts and management.

Accordingly the description of this study is structured as follows. First we outline earlier real-life process mining case studies in Section 5.1.1. Then Section 5.1.2 elaborates on the followed research methodology which is applied within a financial services organization in Section 5.1.3. Finally, we formulate conclusions in Section 5.1.4.

5.1.1 Related Work

In the literature, contributions to the process mining field are most often theoretical in nature, with a large dominance of studies related to the development of knowledge discovery techniques [10, 32, 67, 126, 127, 131, 139]. Only few articles on practical applications have emerged in the last decade. Our work complements the theoretical approaches and integrates different perspectives to search for inefficiencies within organizations. Our findings are validated with a case study in which several existing process mining techniques are applied to a data set originating from the back office process of a financial services company. Although practical process mining cases in the literature are rare, real-life applications are essential to prove the usefulness of Process Mining in practice. Ten relevant papers are listed in Table 5.1 where Process Mining is used as a practical tool to help organizations understand their business.

The first real process mining application study was described by van der Aalst et al. [127]. The authors verify the applicability of existing process mining techniques in a real-life industrial case. The purpose of this study is similar to ours in the sense that a real-life event log is analyzed according to the process, organizational and data perspective. This study had a revolutionary character since at the time it was published, similar things had never been tested before. It was shown that Process Mining techniques are a feasible approach for analyzing business processes, although some limitations were pointed out. Foremost, only the actually logged events can be used for analysis as well as the fact that privacy issues might restrict the possible outcomes of a process mining analysis. In contrast to [127], the most promising application areas of Process Mining are situated within the services industry. This is mainly due to the fact that the strengths of Process Mining lies in the analysis of human-centric behavior which typically results in higher levels of deviations, exceptions etc.

A first domain for which several case studies are present is public services. Alves de Medeiros et al. [10] took an interest in four processes from Dutch municipalities to test the robustness of the Genetic Miner algorithm. Song and van der Aalst change the focus of analysis towards organizational aspects in [121]. Also in this study, a municipality process is used for demonstrating the practical application of the proposed techniques.

Rozinat et al. [112] conducted two more studies in the public services domain. Both studies focus on the discovery and quality of simulation models. In their analyses, theoretical concepts of process mining techniques are tested on a real-life event log, integrating multiple perspectives into a comprehensive simulation model. In our work we use the different perspectives to search for inefficiencies in each dimension instead of merging the three perspectives into one discovered simulation model as done by Rozinat et al. The last two real-life case studies

Author	Domain	Year	Event log details			
			# cases	# activity types	# events	# originators
van der Aalst et al. [127]	Public services	2007	14.279	17	147.579	487
Alves de Medeiros et al. [10]	Public services	2007	4 event logs:			
			35	14	-	-
			100	17	-	-
			358	11	-	-
			407	19	-	-
Mans et al. [89]	Healthcare	2008	627	376	-	-
Song and van der Aalst [121]	Public services	2008	570	9	3.023	109
Jans et al. [76]	Financial services	2009	10.000	7	62.531	290
Rozinat et al. [89]	Public services	2009	2 event logs:			
			363	5	1.817	13
			570	10	6.616	110
Rozinat et al. [110]	Wafer scanner industry	2009	24	720	154.966	-
Goedertier et al. [62]	Telecom industry	2011	17.812	42	80.401	-
Rebuge and Ferreira [105]	Healthcare	2011	627	376	-	-
van der Aalst et al. [140]	Public services	2011	2 event logs:			
			796	8	5.187	-
			1.882	-	11.985	-

Table 5.1: An overview of practical applications of process mining in the literature

performed in this domain are presented in the research of van der Aalst et al. [140]. This paper introduces a new approach to the current process mining techniques. The discovered process models are used as an extension to forecast the completion time of running instances. In both real-life cases the data is split up fifty-fifty to a learning set and test set with the main focus on the quality of the time predictions.

Also in the private sector, the applicability of process mining techniques has been studied. Goedertier et al. [62] apply different process discovery techniques within the telecom industry. It was found that applying these techniques to the highly flexible process at hand created severe challenges for both knowledge discovery as well as knowledge evaluation. Since this study entails an extensive empirical evaluation, it can be considered as a key contribution to the literature on applying Process Mining to real-life data.

Furthermore, the healthcare industry can benefit a lot from Process Mining as well. Mans et al. [89] and Rebuge and Ferreira and [105] demonstrate how process mining can be applied to complex care pathways. Their results reveal the possibility of uncovering understandable models from large groups of patients in order to track deviations from guidelines.

A somewhat rare application in a manufacturing context is presented by Rozinat et al. [110]. The study focuses on the conformance aspect in the wafer scanner industry. The paper points out that in today's processes vital information is missing for compliance purposes, but that in the future more audit data will be available. This observation is further explored by [76]. In this study, it is shown that internal fraud can be investigated by making use of process mining techniques such as the LTL-checker in ProM[1]. Finally, van der Aalst et al. [142] used Process Mining to explore the process model and business rules that are the subject of an auditing task. A disadvantage of this paper is that it predominantly focuses on the theoretical side as compared to practical implications.

In conclusion, this section shows that Process Mining can be successfully applied in a wide variety of practical situations. However, there are still a lot of unexplored paths and the need for more real-life case studies providing evidence for the effectiveness of Process Mining is obvious.

5.1.2 A Methodological Framework for applying Process Mining in Practice

The objective of this case study is to demonstrate the usefulness of Process Mining in practice and as such to provide a suggestion on how Process Mining can be applied in real-life. It has been found that many of the proposed algorithms have difficulties in dealing with real-life event logs since these logs typically exhibit much less structured process behavior [62, 68, 149]. Today, this remains one of

[1] http://www.promtools.org/prom5/

the most important challenges for process mining research. As such, so as to apply process mining techniques in practice, guidelines should be put forward on how to conduct such a case study. Accordingly, this study proposes a methodology framework in line with earlier works [19, 105]. Our Process Mining Methodology Framework (PMMF), as depicted in Figure 5.1, emphasizes the early phases in such an analysis, i.e. data preparation and exploration. Furthermore, our framework acknowledges the multi-faceted analysis that is often required, especially when analyzing highly flexible business processes where the underlying information system allows a lot of freedom to its users. Because of this flexibility, it is typically found that the logged data is less structured and much more difficult to analyze. The demonstration of the application of our framework in Section 5.1.3 can help process miners to overcome different issues encountered during a process mining analysis.

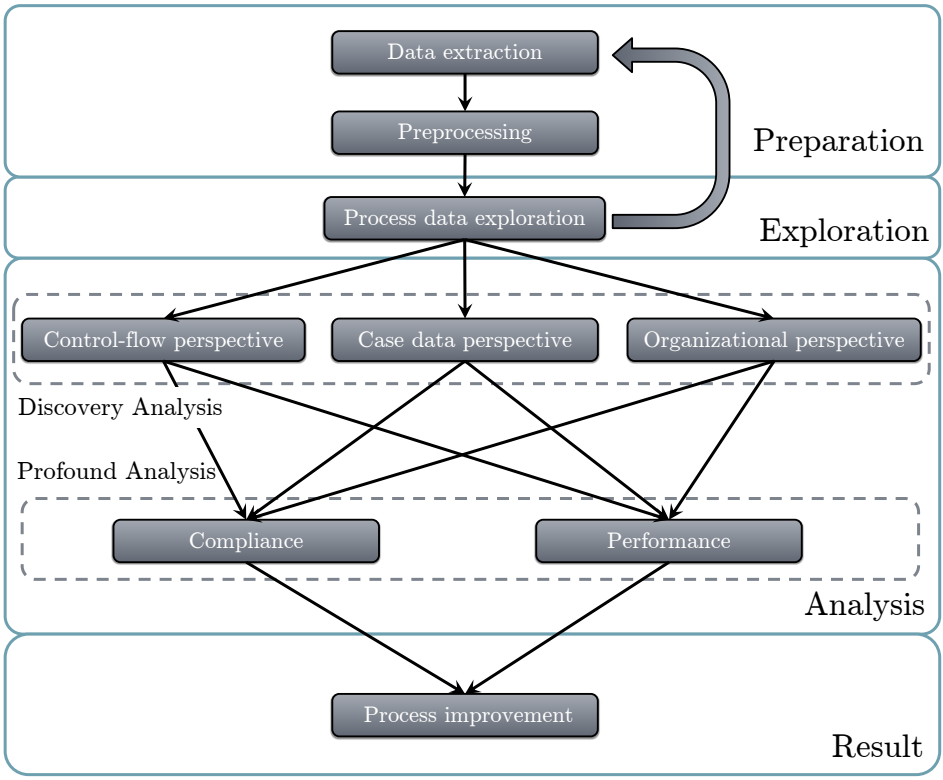


Figure 5.1: The Process Mining Methodology Framework

The framework consists of four major building blocks: preparation, exploration, analysis and results. Before any analyses can be performed, data must

be collected and prepared. The first component in the *preparation* phase is the data extraction part. A first challenge crops up when the process scope and the time frame are to be determined. The goal is to extract data from the bulk of information stored in the system that is relevant for the analysis. There is a clear trade-off between a too wide and a too narrow process scope. When too much data is extracted two or more processes can get mixed up, giving a process model that is hard to interpret. If less data is incorporated in the analysis, a straightforward interpretation can be obtained, but the danger of missing part of the knowledge increases. The problem of scope determination also turns up in the time dimension. A time frame that is too wide can cause an overflow of data, causing existing process mining techniques requiring large computation times and often resulting in incomprehensible results. On the other hand, if the process fluctuates strongly because of seasonality for example, a small time frame may not be long enough to include such patterns. Overall, process scope and time frame definition are highly industry dependent but once the appropriate settings are agreed upon, effective preprocessing can begin by converting the data into a workable format for analysis.

The next phase is process data *exploration*, where a first impression of the data is gathered. For this purpose, a lot of statistical information can be assessed. Also, with the availability of different control-flow mining algorithms, a number of initial visualizations of the process can be obtained. Relying on these exploration steps, business experts can iteratively improve the process scope and time frame in order to ensure that the input data is suitable for further analysis. The feedback loop for the adjustment of both process scope and time frame is imperative and therefore explicitly depicted in Figure 5.1.

After agreeing upon a satisfactory data set according to time and scope, a more profound *analysis* can be initiated. The analysis phase is subdivided into two major segments: the basic discovery analysis and the more detailed compliance and performance analysis. In the discovery phase, we distinguish a control-flow, an organizational and a case data perspective on the data. The control-flow perspective includes the analysis of activity sequences within the business process. Furthermore, the data can be explored from an organizational point of view, for instance by investigating the teams involved in the process. Finally, it is often valuable to explore the underlying data elements of process executions in order to discover particular patterns.

Typically, a discovery analysis will highlight different points of interest for which a more profound assessment is appropriate. The PMMF considers two different types of profound analyses: performance and compliance analysis. In the case study presented below, management was mainly interested in case throughput times. On the one hand, performance analysis can be carried out in general for the whole set of process executions. On the other hand, it is typically worthwhile

to explore performance in more detail. Subsets of traces are then investigated in order to examine the impact of control-flow or other characteristics of these process executions. The profound analysis also consists of compliance analyses which can be of interest to the organization. Compliance verification is about validating whether the reality is consistent with expected or required process behavior. In Section 5.1.3, compliance issues are thoroughly investigated because process inefficiencies coincided with deviating behavior with respect to management's expectations.

Finally, the *result* phase is the closing stage of the PMMF. The outcome of the analyses will be a valuable starting point for process improvement or even process reengineering steps in order to optimize the business process at hand. Management can define new goals and measurements based on the new insights obtained with Process Mining in order to resolve for instance identified process inefficiencies.

5.1.3 Case Study

With the purpose of this study being the demonstration of the usefulness of process mining analyses in practice, we describe a case in the financial services industry. This industry is of main interest for Process Mining since there exist plenty of human-centric business processes for which the analysis of event logs proves especially worthwhile. The case at hand involves a large Belgian insurance company. Products include life and non-life insurances and retirement savings, which are mainly offered through a large network of brokers.

Context

The core information system underlying the insurance company's back office can be best described as a Document Management System (DMS). As soon as a physical document arrives at the company, being an e-mail, a claim, an application offer, etc., it is digitalized and added to the DMS. Each document receives a document ID and document type before it is sent to a team for further processing. From then on, the system keeps track of every step taken during the document's life cycle by recording which actions were taken by which team at which moment in time. These data are the cornerstone for the process mining case study outlined below.

Although plenty of process-related information is available, the company had no clear idea of how the back office processes actually look like, nor could it provide accurate performance estimations for the different document types. In the next section the process of finding the answers to these questions using the Process Mining Methodology Framework is explained.

Data preparation and exploration

The first crucial phase in any process mining analysis consists of preparation and exploration of the available process data. All the event information needed to conduct a process mining analysis was present in the DMS of the company. In a first phase, these data were extracted from the DMS and converted into a standard event log storage format, in this case MXML.

Process data exploration and scope adjustment. In a next step, the extracted data was imported into the analysis tool ProM, where an early inspection could start. A major challenge encountered was the substantial size of the data set causing difficulties for both data processing and data analysis. A first important reason why the data was difficult to work with was the absence of almost any restriction on the possible execution paths within the system. It was found that the DMS was nowhere near the more structured process behavior typically found in for example BPMSs. Due to the unlimited number of possible process executions, any typical control-flow process model rendered incomprehensible. As a consequence, it proved very difficult to mark out the boundaries of the process under investigation in the larger DMS. Therefore, the data scope had to be fine-tuned multiple times, going through the scope adjustment loop after each inspection, until a satisfactory data set was obtained.

In consultation with the insurance company's management, the final event log consisted of the execution paths of all the proposals dealt with in a six-month period. Because of the typicality of the data extracted from the DMS, it turned out that the event data could be decoupled into three different event logs according to three different information perspectives. The first log corresponded to the control-flow, where each case (being a document) followed a certain sequence of activities. A second log was constituted from a performer viewpoint: each case now followed a certain sequence of teams. More specifically, each event corresponds to a change of hands, so that each time a document was forwarded, that event was included. The last log was composed from a case data view based on the attribute document type. This was useful, since an incoming document receives a preliminary type, which can be further specified or changed during its life time. In the same way as the former log, the document type flow can be tracked per process execution.

The construction of multiple event logs from the same data set according to three different perspectives resulted in an atypical way of working. Usually, an event log is centered on capturing control-flow activity events which are performed by a certain originator and to which a number of case data elements can be added. All three analysis perspectives can be investigated by analyzing this single event log. However, in this case we constructed three different event logs according to the different perspectives. The reason why we followed a different strategy stems

from both the specificity of the data and the expectations of the organization's executives. After several discussions, it appeared that both the team flow and the document type flow represented the business process best. Accordingly, it appeared valuable to make available control-flow mining techniques with respect to the teams and the document types.

Discovery analysis

According to the methodological framework, the three discerned event logs are firstly analyzed in an exploratory way in order to find interesting observations for further analysis.

Control-flow event log: business activities. The control-flow based event log contained a total of 44.880 cases (documents) with fifteen real business activities. A discovery analysis typically starts with the visualization of the underlying process model. The visualization for this event log is strongly complicated by the fact that the event log consists of 4.494 distinct process executions (DPI), as illustrated in Table 5.2. The complexity is even further emphasized by the fact that 3.345 instances are unique activity sequences.

One method to limit the variety in process behavior is to narrow the scope to more frequent demeanor. For this case study the log was filtered based on process instance frequencies. A process instance frequency is interpreted as the number of times a same sequence of activities is traversed by different process instances. Applying a frequency threshold of fifty, a total 34.769 cases or 77, 47% of all cases were withheld. On this filtered data set, HeuristicsMiner [153] was performed in order to visualize the process as in Figure 5.2. This process model covers 667 frequent process executions. Accordingly, this model only conforms to mainstream behavior and thus excludes exceptions and infrequent cases that might be interesting as well.

From a business perspective, no particular conclusions can be drawn from the control-flow analysis. Although the analysis indicates a strong behavioral freedom accorded by the information system, this freedom is often necessary and efficient. More stringent conditions on possible activity sequences might be introduced in order to reduce the level of self-determination of the users, but there is no guarantee that the process itself could be improved significantly in this way. In order to tune the analysis to business interests, the attention shifted to features business users work with on a daily basis: documents and other teams. In next paragraphs the corresponding perspectives are explored.

Organizational event log: team routing. The process data can also be examined from an organizational perspective, where a flow represents the sequence

	Frequency threshold	Event log characteristics			Heuristic net characteristics	
		# PI	# DPI	#AT	# Arcs	# Activities
Control-flow log	0	44.880 (100%)	4.494	15 activities	209	16
	50	34.769 (77,47%)	667	13 activities	69	14
Team flow log	0	44.880 (100%)	1.222	59 teams	498	61
	50	40.406 (90,47%)	77	30 teams	165	32
Document flow log	0	44.880 (100%)	2.027	189 documents	209	191
	50	39.396 (87,78%)	41	22 documents	69	24

Table 5.2: Characteristics of the event logs applying trace-based frequency filtering for the different perspectives - the number of process instances (# PI), the number of distinct process instances (# DPI), the number of activity types (# AT)

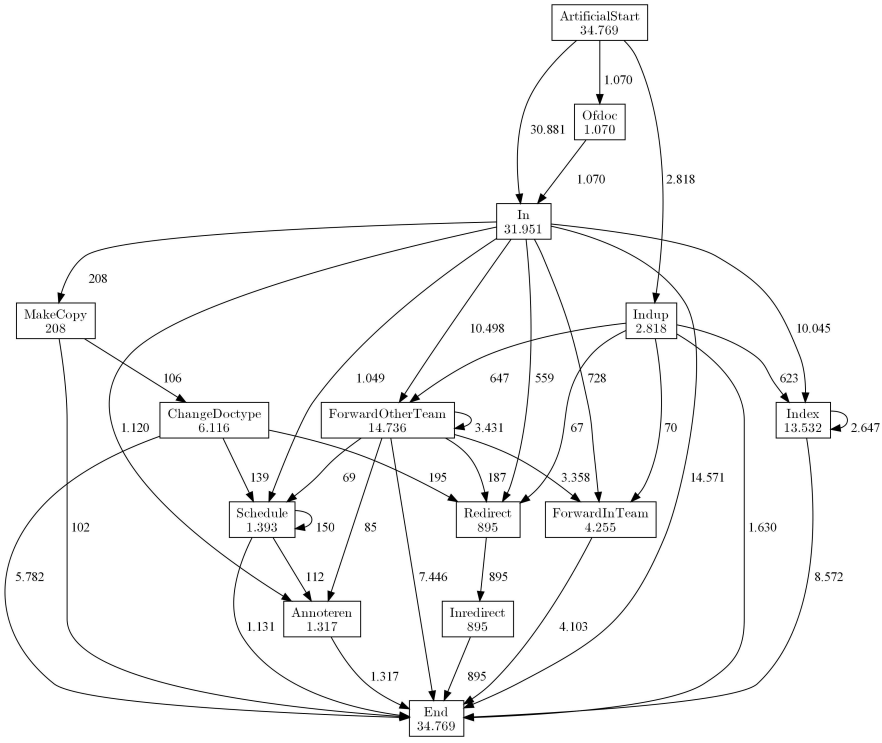


Figure 5.2: Visualization of the heuristics net for the control-flow log with a frequency threshold of 50

of teams that worked on the same document. A first attempt to visualize the team flows resulted in a completely incomprehensible model. It reflected the complexity of the data in the log: 1.222 different paths could be followed by 59 teams, represented by 498 arcs in the heuristic net (see Table 5.2). To allow for the extraction of useful knowledge from the log, the amount of team paths needed to be reduced in a similar fashion as described in Section 5.1.3. Focusing on frequent behavior, a general view on how teams were interconnected could be obtained. In a similar fashion as with the control-flow perspective the event log was filtered for team paths that were traversed by at least fifty different document instances. The result was an involvement of only 30 teams in 77 possible paths presented by the heuristic net in Figure 5.3. Although this might imply a drastic cut in the data, the opposite appeared to be true: more than ninety percent of all the log traces were retained after filtering.

Looking more closely at the team flow process model, it can be pointed out that there are three teams operating on cases more than 10.000 times: In, Central and A1. First of all, the In-team received part of the incoming mail and forwarded the documents to a specific team for further processing. If there was confusion about which team was next, the document was sent to the Central-team. This team, involved 15.415 times in the flow, was responsible for those forwarded mails and for most other incoming documents. It acted as a central administration point and firstly defined the document type. Afterwards the documents were forwarded to a team specialized in the document type at hand. When a document was wrongly forwarded, most likely the other team sent it back. A last team that had a high involvement (10.025 times) was the A1-team which was mainly due to the fact that this team could treat multiple document types.

By discussing these initial findings with the business users, a first possible inefficiency was identified. Several teams stated that they were burdened by reiterated handovers, being documents that were sent back and forth between teams. Since this issue was confirmed by our analysis (see Figure 5.3), this behavior is further investigated in Section 5.1.3.

Case data event log: document types. Finally, an exploratory analysis of the document types was carried out. As such, we investigated the sequences of the document type changes. Due to the infeasibility of displaying the model for the complete event log, a filter was applied similar to former perspectives to visualize only frequent document paths.

The results of the filtering were striking since the number of document types decreased from 189 to only 22 and the total of different possible paths declined from 2.027 to 41. This signifies that the filtering method applied was highly proficient in separating frequent and infrequent behavior. This is confirmed by the fact that almost 90 percent of the logged data formed the input for the resulting

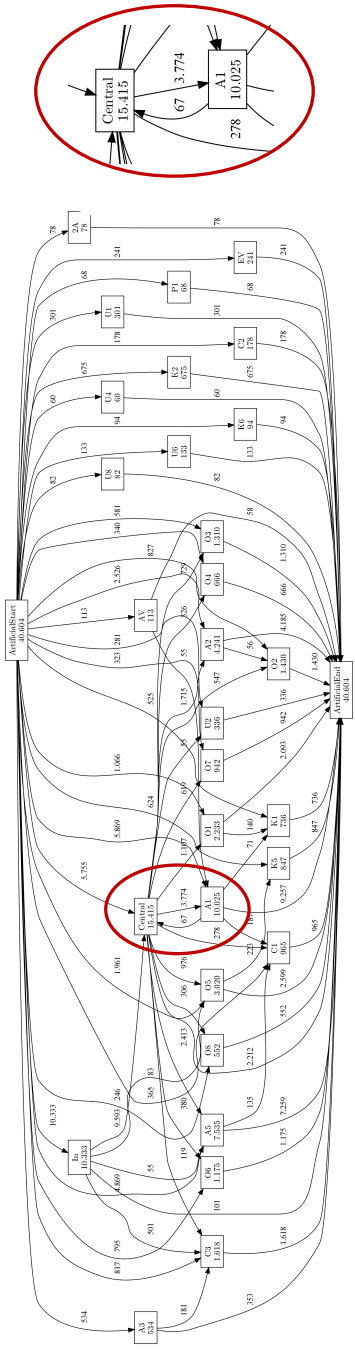


Figure 5.3: Visualization of the heuristics net for the team flow log with a frequency threshold of 50. The enlargement shows one of the many reiterated handovers between teams

process model in Figure 5.4. So even though the system exhibits a large amount of freedom, a large amount of behavior is captured by a small amount of possible document type sequences. This emphasizes the relevance of analyzing the most frequent document paths.

An analysis of the discovered model highlighted the issue of wrong document classification. For instance, certain documents change from a private proposal to a business proposal or vice versa. In consultation with business experts, this problem was considered for more profound analysis.

Profound analysis: tracking process inefficiencies

In the profound analysis, the investigation was taken a step further, digging deeper into the data to come up with interesting insights. Because of the main focus on performance in terms of throughput time, it was decided to create a benchmark event log to better judge unwanted process behavior.

Performance analysis: creating a benchmark. In order to assess the impact of the compliance issues explained in the next sections, a benchmark log was created first. It represents standard behavior corresponding to very frequent, normal process executions. The analysis started from the document perspective since this was the most intuitive view for management. Former conditions were fulfilled when looking at the six most occurring document flows depicted in Figure 5.5. After isolating the instance IDs corresponding to these conventional paths, the corresponding documents were also retained in the other two perspectives. Questions such as "which teams treated those documents?" and "what business activities are usually performed in a standard process?" were answered.

The histogram in Figure 5.6a displays the throughput time distribution of the benchmark event log. The x-axis depicts the throughput time, calculated in working days of nine hours. On the vertical axis the frequency in number of instances is shown. The mean time a document resided in the system before it was terminated is 8,26 working days, with a standard deviation of 8,5 days. In the final analysis sections, we will employ the characteristics of this benchmark event log as a point of reference to evaluate bottlenecks and other process inefficiencies.

Compliance analysis. Verifying compliance is an important step towards process improvement. Compliance can be assessed by verifying strictly defined rules and regulations, but also by contrasting management's expectations with the actual process behavior. Both types of compliance analysis feature in this section. Firstly, a formal analysis was executed based on a business rule concerning document classification. Next a more informal analysis looked at whether management expectations in terms of reiterated team handover were confirmed.

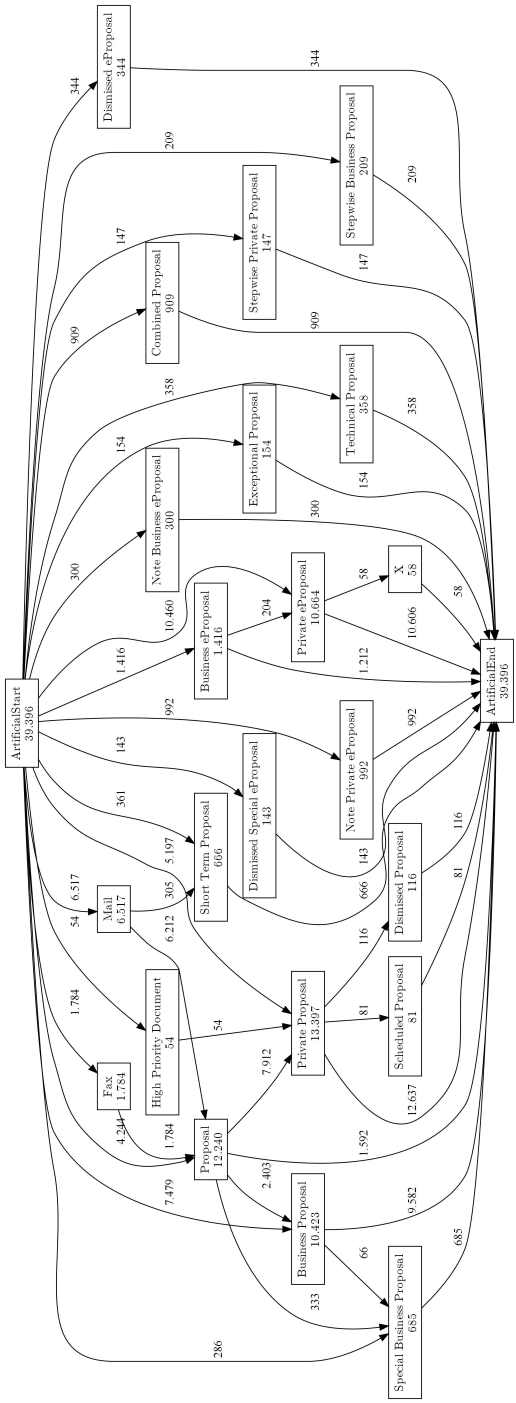


Figure 5.4: Visualization (heuristics net) for the document type event log with a frequency threshold of 50

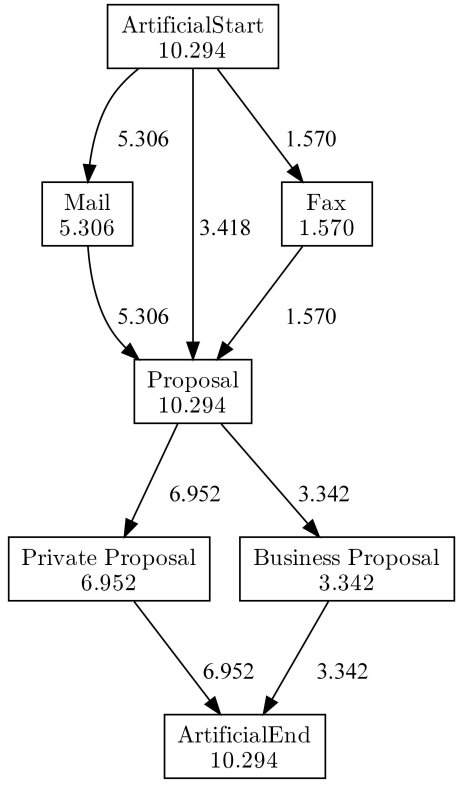


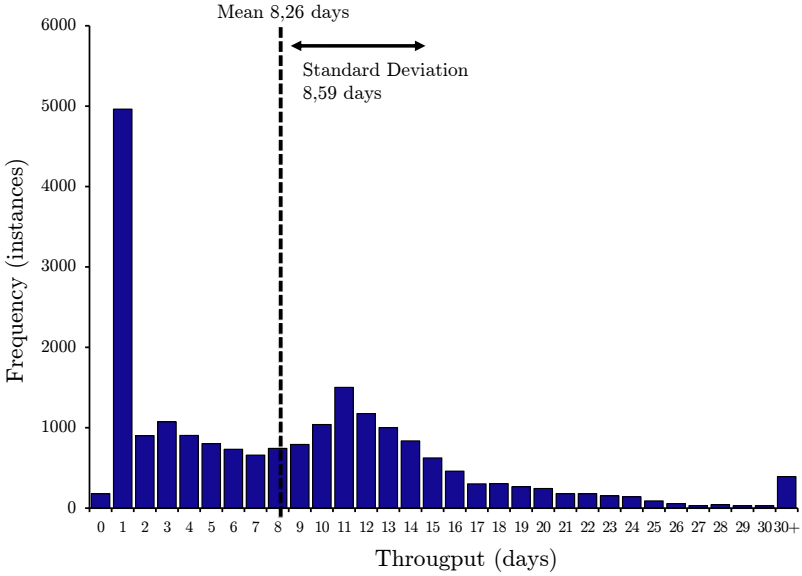
Figure 5.5: Visualization (heuristics net) of the six most frequent document type sequences

Wrong document classification. An example of a business rule in the insurance company's back office is the requirement that certain document types were not allowed to appear in the same document type sequence. However, the data contained several violations against this rule. Often, documents were originally classified as a business proposal but later on in the process rectified to a private proposal or the other way around. Note that due to the application of thresholds by the HeuristicsMiner algorithm, misclassification paths are not directly detectable in the process model in Figure 5.4. To this, it should be added that some paths are present in the model but difficult to identify because of the use of (hidden) AND-splits. This feature makes a heuristic net not straightforward to interpret by business users and makes the discovered process models less appropriate for discussion with organization executives. This is also confirmed by Mendling et al. [93] who investigate the influence factors of process model understandability. Better visualization techniques are therefore requisite to avoid wrong interpretations and improve the understandability for business users.

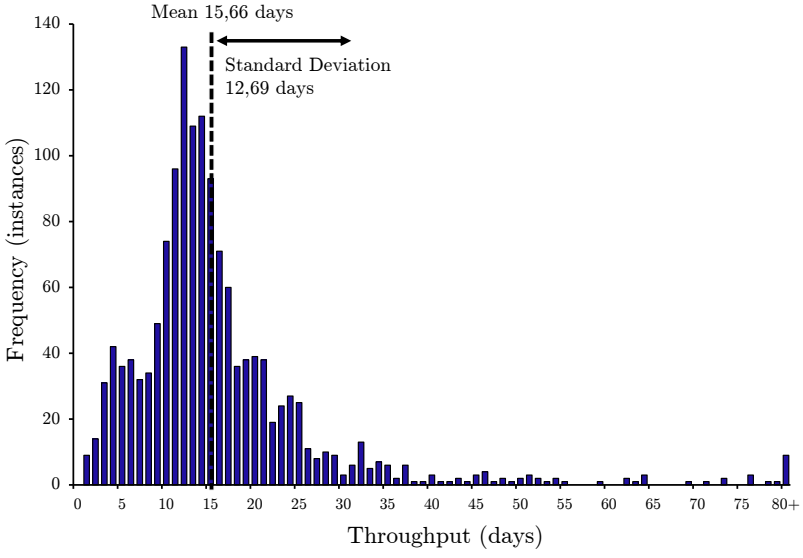
To estimate the impact of this inefficiency, all document type traces in which both a private proposal and a business proposal occurred, were filtered out. There appeared to be 1.422 cases in the data fulfilling this condition. After conducting a performance analysis on traces with one or more wrong document classifications, it was shown that it took on average 7,41 working days longer to complete such a document as compared to a normal trace (Figure 5.6b). This is a significant difference, especially because other faulty document type reclassifications reside in the process as well. Since a similar throughput time problem can be expected, improvement actions seem indispensable in this area. As such, the analyses can provide important decision support in order to shape a process reengineering plan.

Reiterated team handover. At the insurance company, teams complained about a high amount of documents that were forwarded and then sent back to the sender. It was observed that in 841 cases, reiterated handover between teams existed. Therefore, we decided to investigate the influence on the efficiency of the process in terms of throughput time. In order to separate process executions with repeated forwarding, the analysis started from the organizational log. By using the LTL-checker in ProM, we were able to identify and separate those process executions for which a particular team appeared more than once. An example of this reiterated handover between teams is shown on the right-hand side of Figure 5.3 for the teams Control and A1.

Afterwards, the performance information for all combined reiterated handover cases was looked at. Our presumption was confirmed: a reiterated handover trace endured longer in comparison with standard behavior (Figure 5.6c). On average, it took twice as long as a normal execution path (17 hours compared to 8 hours), even worse than the document classification problem. Although this behavior

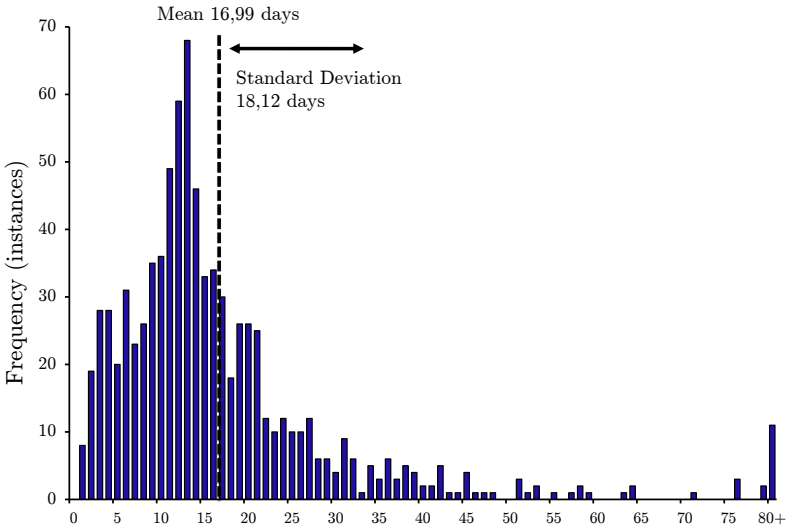


(a) Benchmark event log



(b) Document classification problem

Figure 5.6: Histograms showing performance problems related to document misclassification and reiterated handover



(c) Reiterated handover problem

Figure 5.6: Histograms showing performance problems related to document misclassification and reiterated handover

represents only 2 percent of the log, it can be interesting to look at the cause of this behavior, since it impacts the throughput in a very negative way.

Results: process improvement measures

Ultimately, the results of the case study were considered remarkable by the organization's management. By contrasting actual behavior as registered in the logged data with expectations and requirements, different insights and guidelines for process improvement could be formulated. Firstly, we observed that the quality of a process mining study can only be as good as the quality of the input data. For instance, clear verb-object names for denoting activities prove very helpful for interpretation of the data. Furthermore, for enhanced performance analysis, both start and end timestamps of activities should be kept track of. According to these findings, a number of recommendations were proposed to improve and guarantee data quality.

Secondly, the opportunity to work out a business case for the introduction of OCR (Optical Character Recognition) technology was identified. Since the document misclassification appeared to be an important source of process inefficiency, OCR might resolve this issue. By improving this manual process to a fully au-

tomated scanning and classification process where the type of the document is recognized based on keywords, significant process performance gains can be expected.

Finally, the exposure of multiple inefficiencies offers a great opportunity for management to make employees aware of them and improve the business process by better training and guidance. For instance, the downside of frequent reiterated handover could be emphasized in order to reduce process inefficiency.

5.1.4 Conclusion

In this study we addressed the issue of the applicability of Process Mining in real-life environments. We acknowledge the importance of applied process mining studies, especially since their conclusions differ strongly from the important work that has been done on the theoretical side. However, the related work section indicates that the issue of applying process mining techniques in practice should receive more attention, also in academic literature. Accordingly, we provide a methodology framework structuring the process of a real-life analysis and illustrate its usefulness in a multi-faceted case study, situated in the financial services industry. The starting point of analysis was process data extracted from a Document Management System as implemented by a large Belgian insurance company to support its back office processes.

Our case study shows that an important element of process mining analysis is the data preparation and data exploration part. More specifically, the information feedback loop between process analysts and business experts proves to be a critical success factor. This two-way communication is essential to discover useful business insights based on the mathematical techniques of the process mining field. Accordingly, the feedback loop allows for improvement of both the process scope and the quality of the data. A second observation is the usefulness of combining several perspectives to get a fuller understanding of the process, which is entirely captured by the third phase of our methodological framework. First a discovery analysis can indicate what perspectives are interesting from a management point of view. This enables a more focused and more valuable scope for further analysis. Secondly the integration of several perspectives in an innovative way exploits the capability of searching inefficiencies instead of solely discovering process models. Subsequently, based on the outcomes of our analysis, it is found that Process Mining can be an effective and efficient technique to deal with organizational challenges. Moreover, a process mining analysis on real-life event logs can be the starting point for the involved management to formulate specific measures to improve the business processes. Another conclusion is the applicability of process mining techniques on structured and less structured information systems. In the case of a less process-based information system such as a Document Management

System, Process Mining allows to obtain a model of the actual process based on the data in the event log. Process Mining proves especially useful to contrast expected behavior with actual behavior as reflected in the data. Since more flexibility is an important characteristic of information systems that are process-aware but not entirely process-based, Process Mining is an ideal means to discover how the system is used and where process inefficiencies should be countered.

Furthermore, the case study uncovers several limitations of the application of currently available process mining techniques on real-life data. First of all, the greatest strength of Process Mining, its reliance on the actual data, is a weakness as well. Today, in many organizations, logging infrastructures are imperfect and business activities executed outside the system cannot be taken into account for analysis. Also, process analysts must rely on the quality of delivered input data. Profoundness and correctness of results are greatly determined by the quality of the process data. Secondly, process mining techniques still struggle with substantial amounts of unstructured data. Highly complex processes, which can be expected in highly flexible environments such as financial services back offices, are a definite challenge. As explained in this study, the use of relevant filters can be a solution to extract important knowledge from such event logs. However, we think that future research in this area is required to enhance and adapt process mining techniques to real-life environments.

5.2 Getting a Grasp on Clinical Pathway Data: An Approach Based on Process Mining

Since healthcare processes are pre-eminently heterogeneous and multi-disciplinary, information systems supporting these processes face important challenges in terms of design, implementation and diagnosis. Nonetheless, streamlining clinical pathways with the purpose of delivering high quality care while at the same time reducing costs is a promising goal. In this case study, we propose a methodology founded on process mining for intelligent analysis of clinical pathway data. Process mining can be considered a valuable approach to obtain a better understanding about the actual way of working in human-centric processes such as clinical pathways by investigating the event data as recorded in healthcare information systems. However, capturing tangible knowledge from clinical processes with their ad hoc and complex nature proves difficult.

5.2.1 Introduction

Worldwide, the healthcare sector goes through a major reform. This has many reasons. First, the costs of healthcare are rising up to 15% in the United States (US) and close to 10% in Europe [101]. This is due to the increasing needs of a greying population, but also due to technological and pharmacological innovations that are really widening the possibilities for diagnosis and treatment. Second, there is a shift in the role of patients, going from a more passive role into a role of active consumers of care. Patients want to be informed and involved. Third, there is growing attention to quality and safety. The main drive comes from the *to err is human* report from the Institute of Medicine[2] [78]. This report indicated that as many as 44.000 to 98.000 US citizens die in hospitals each year as the result of medical errors. Even using the lower estimate, this would make medical errors the eighth leading cause of death in the US - higher than motor vehicle accidents (43.458), breast cancer (42.297) or AIDS (16.516). The report was publicly discussed in the Senate and was the start of an overall hospital reform. Most important in the discussion is that people are not blamed - *to err is human* indeed - but that the focus should be on improving the system.

An excellent way to do this is learning from past experience. Currently, it can be observed that with the growing implementation of integrated healthcare information systems, vast amounts of data are becoming available about the actual way of working in clinical pathways. These data form the cornerstone of this study. Accordingly, the notion of a clinical pathway is a crucial element. The terminology has its origins in methodologies such as PERT (Project Evaluation and Review

[2] http://www.iom.edu

Technique) and CPM (Critical Path Method), but transformed into "clinical" instead of "critical" pathways because of the very specific nature of healthcare. When clinical pathways were developed in the mid eighties, their major focus was on reducing the length of stay (LOS) of patients. The first systematic use was found in the New England Medical Center in Boston (USA) in 1985 as a response to the introduction of Diagnosis Related Groups (DRGs) in 1983. The actual focus is mainly on improving the quality of care. Clinical pathways are formally defined as *schedules of medical and nursing procedures, including diagnostic tests, medications and consultations, designed to effect an efficient, coordinated program of treatment* by the National Library of Medicine [99], which is a very narrow definition. The European Pathway Association[3] broadened this definition to *a complex intervention for the mutual decision making and organization of care for a well-defined group of patients during a well-defined period.*

In this study, we propose an approach for deriving useful insights from clinical pathway data by making use of process mining techniques. The main contribution of this study is the development of solution strategies for dealing with the extremely unstructured nature of clinical pathway data.

5.2.2 Process Management in Healthcare Organizations

Before elaborating on the data set and the analysis methodology, the research is situated in the context of its related work.

Process Aware Healthcare Information Systems

A fair share of information systems implemented in healthcare organizations can be described as process aware. This is because process aware information systems not only encompass traditional workflow management systems, but also include systems that provide much more flexibility. Accordingly, once an information system can be described as having an explicit notion of the process it supports, it can be described as process aware [48]. A such, many healthcare information systems fit within this definition. Since clinical pathways are inherently heterogeneous and multi-disciplinary in nature, the goal of IT support for healthcare processes is not to control the course of the process entirely, but to assist healthcare professionals by reducing cognitive overload and improving the basis for their decisions [84]. In this way, process orientation can be considered as a beneficial approach towards streamlining clinical pathways with the purpose of delivering high quality care while at the same time reducing costs [12]. While business process support for structured processes (e.g. manufacturing, logistics) has always been an important research topic, the growing importance of service organizations such as healthcare

[3] http://www.e-p-a.org

has triggered the need for different approaches towards business process support [83, 106]. Because of the human-centric nature of such processes, they contain much more flexibility, alternative routings, loops, human judgement and variability than traditional business processes. Accordingly, the analysis of the actual way of working by making use of the data captured by healthcare information systems is promising. However, traditional business process analysis techniques come short in realizing this goal and therefore a novel analysis methodology based on process mining is proposed.

Process Mining in Healthcare

Process mining techniques have been applied in a healthcare context. A first study by Mans et al. [89] shows how different process mining techniques such as HeuristicsMiner, social network analysis and dotted chart analysis allow for obtaining insights into care flow data. Another study by Rebuge and Ferreira [105] also describes a methodology for the analysis of business processes in a healthcare environment. The methodology consists of seven phases with its main asset being the application of sequence clustering techniques. Further, Bose and van der Aalst [17] propose the use of fuzzy mining and trace alignment for investigating clinical pathway data. Finally, Caron et al. [28] demonstrate the applicability of various process mining techniques to healthcare data by adopting both a department and a treatment based focus. Our study differs from previous studies because it shows the benefits of both a drill up and a drill down perspective on the data relying on control-flow discovery with the Fuzzy Miner and networked graph visualizations.

5.2.3 Description of the Clinical Pathway Data

The data set concerns real data of a gynecological oncology process at the AMC hospital in Amsterdam, The Netherlands. It was first used in [89], but recently the data set was made publicly available (doi:10.4121/uuid:d9769f3d-0ab0-4fb8-803b-0d1120ffcf54) for the first Business Process Intelligence Challenge (BPIC'11). The data contains 150.291 events of 1.143 patient treatment processes related to individuals diagnosed with cancer pertaining to the cervix, vulva, uterus and/or ovary. Each case in the event log corresponds to a single patient and as such, the data presents a wide variety of care activity sequences. In the remainder of this section, the three most important dimensions of the data are outlined.

Diagnosis. Each case in the data set contains information on the type of disease the patient is diagnosed with. The related attributes are denoted *Diagnosis code* and *Diagnosis*. The data presents a total of eleven different diagnosis codes (e.g. M11, M12, 823, etc.). *Diagnosis* is a textual description which specifies the diagnosis code, taking values such as "adenocarcinoma stage Ia" or "clear cell carcinoma".

Case ID	Event name	Dept.	Timestamp	Diagn. code	Diagnosis	Treatm. code	Age	...
0	1st polyclinic consult	Radiotherapy	03/01/2005	M13	cervical malign.	23	33	...
0	administrative reg.	Radiotherapy	03/01/2005	M13	cervical malign.	23	33	...
0	gynec. cost assign.	Nursing ward	03/01/2005	M13	cervical malign.	23	33	...
0	ultrasonography	Obstr.&gyn. clinic	03/01/2005	M13	cervical malign.	23	33	...
0	1st consult	Nursing ward	03/01/2005	M13	cervical malign.	23	33	...
...	...	...	...	...	...	...	...	...

Table 5.3: Excerpt of the event log

Code	Region	Example diagnoses	# cases
M11	vulva	squamous cell carcinoma, borderline malignancy	176
M12	vagina	squamous cell carcinoma, adenocarcinoma	22
M13	cervix	squamous cell carcinoma, malignant neoplasms	368
M14	corpus uteri	adenocarcinoma, clear cell carcinoma	145
M15	corpus uteri, myometrium	sarcoma	17
M16	ovary	squamous cell carcinoma, non-epithelial malignancy	235
106	cervix, vulva, corpus uteri, vagina	squamous cell carcinoma, borderline malignancy	298
821	ovary	serous and mucinous squamous cell carcinoma, neoplasms	48
822	cervix	squamous cell carcinoma, adenocarcinoma	131
823	corpus uteri, ovary, endometrium	(serous) adenocarcinoma	16
839	ovary, vulva	serous adenocarcinoma, malignant neoplasms	21

Table 5.4: Diagnosis codes in the clinical pathway data

It should be noted that the data contains up to 16 diagnosis code - diagnosis combinations for a single patient (denoted as *diagnosis code:1 to 16*). Accordingly, a single case might contain different codes. We observe 38 distinct diagnosis code combinations, for instance {M16, 821}. Table 5.4 presents an overview of the diagnosis codes detailing the region, example diagnoses and the number of cases showing this diagnosis code.

Treatment. Next to diagnosis information, one can also find details concerning the treatments of each of the patients. However, in contrast to diagnosis, the data only provides a treatment code and no further information. As such, the treatment perspective is more difficult to analyze. On top of this, there exist 46 distinct treatment codes which form 236 distinct treatment code combinations in similar fashion as the diagnosis code combinations.

Departments. The final important data dimension is organizational in nature, i.e. the departments that are involved in the clinical pathways. Each event in the log contains an attribute "org:group" which denotes the department where the corresponding activity was performed. In the data, one can find 43 distinct organizational units. The most frequently observed departments are depicted in Table 5.5.

Table 5.5: Number of events pertaining to organizational units by frequency

Organizational unit	# events
General Lab Clinical Chemistry	94917
Nursing ward	31066
Obstetrics & Gynaecology clinic	7065
Medical Microbiology	4170
Radiology	3171
Radiotherapy	2233
Internal Specialisms clinic	2146
Pathology	1975
Operating rooms	942
Pharmacy Laboratory	498
Recovery room / high care	495
Nuclear Medicine	281
Special lab radiology	279
...	...

A very specific feature of the data is that events pertaining to certain departments occur in bursts. For instance, sets of blood diagnosis tests performed by the *General Lab Clinical Chemistry* department are often found. Similarly, bursts of

radiotherapy-, nursing-, operating room-related and many other types of events can be observed. This data characteristic will be further employed in the next section.

5.2.4 Analysis Methodology and Results

Our data analysis approach combines two important strategies for extracting tangible knowledge from clinical pathway data. First, drill up is applied in order to get insight into the general behavior of the healthcare process. In a second phase, a drill down approach is described that centers on a certain part of the data.

Complexity of Clinical Pathway Data

The crucial challenge for data analysis in the context of clinical pathways is the complexity of the data. This is because clinical pathways are inherently ad hoc, multi-disciplinary and strongly human-centric. Because of these characteristics, almost every observed clinical pathway is unique, which is also the case for the data set employed in this study. On top of that, the original data set contains 624 different activity types. Further, these activity types as registered in the different departments are not always of the same granularity. A final element that complicates the data analysis significantly is the fact that we can only observe care activities executed within the AMC hospital. However, it is undoubtedly reasonable to assume that other care activities in peripheral hospitals, by GP's, etc. are being executed but not registered in the data. Because of these data complexities, there is a need for versatile data analysis methods. In this study, it is shown how process mining techniques can be used, both in a drill up as well as in a drill down mode. Furthermore, we demonstrate the use of networked graphs for visualizing sets of cases and their respective characteristics.

Drill Up Analysis

Both in [28] and [89], it is shown that the straightforward application of existing process discovery techniques is infeasible for the data at hand. Even the stronger generalization capabilities of Fuzzy Mining [68] prove not very helpful. Therefore, we show how abstraction applied in a data preprocessing step can be beneficial in order to obtain general, but useful insights based on the entire data set.

Data preprocessing. Realizing abstraction in a data preprocessing phase consists of replacing the bursts of events belonging to the same organizational unit by the name of the organizational unit itself. In this way, a clinical pathway in terms of the unique activities performed by different organizational units is transformed into sequences of departments.

A general view using process discovery. As stated earlier, process discovery is the most important asset of the process mining domain. Process discovery is defined as the extraction of control-flow models from event logs. Note that these techniques make use of different process modeling notations (e.g. Petri nets, heuristic nets, fuzzy nets, etc.) in order to represent the discovered model. In this case, we applied the Fuzzy Miner to the transformed clinical pathways. The resulting fuzzy net is depicted in Figure 5.7. Note that the nodes contain a significance value between 0 and 1. Further, the figures on the edges indicate the edge significance and correlation, also ranging between 0 and 1.

With the purpose to increase the comprehensibility, we restricted the visualization to the eleven most frequent departments. Together with the abstraction power of Fuzzy Miner, the discovered graph provides some interesting insights with respect to the gynecological oncology process under investigation:

- A majority of the patients first visit the obstetrics and gynecology department.
- From top to bottom, we can clearly observe the diagnostic-therapeutic cycle characteristic to the majority of care processes. The nursing wards have a pivotal role in between diagnostics and therapeutics.
- From a diagnostic perspective, lab analyses (majorally blood sample tests) and to a lesser extent radiology are essential elements of disease typification in the context of gynecological oncology.
- Despite the fact that the data cover patients with very comparable diagnoses (i.e. gynecological cancers), streamlined clinical pathways cannot be observed, even in terms of involved departments.

Drill Down Analysis

As described in the previous section, drill up is a valuable approach for extracting general knowledge from care process data. Nevertheless, due to the characteristics of the data, intelligent drill down into specific parts of the data is bound to provide even more interesting insights. Hereto, we demonstrate a particular focus on the therapeutic side of the clinical pathways. Since the data provides only limited information on specific treatments (only treatment codes, without any explanation), it is opted to investigate this perspective in more detail.

Focalizing on therapeutic activities. To adopt the focus on treatment, therapeutic activities need to be singled out. By inspection of the cases, it can be noticed that three different types of treatments can be identified: radiotherapy, chemotherapy and surgery. For radiotherapy, the selection of activities was rather

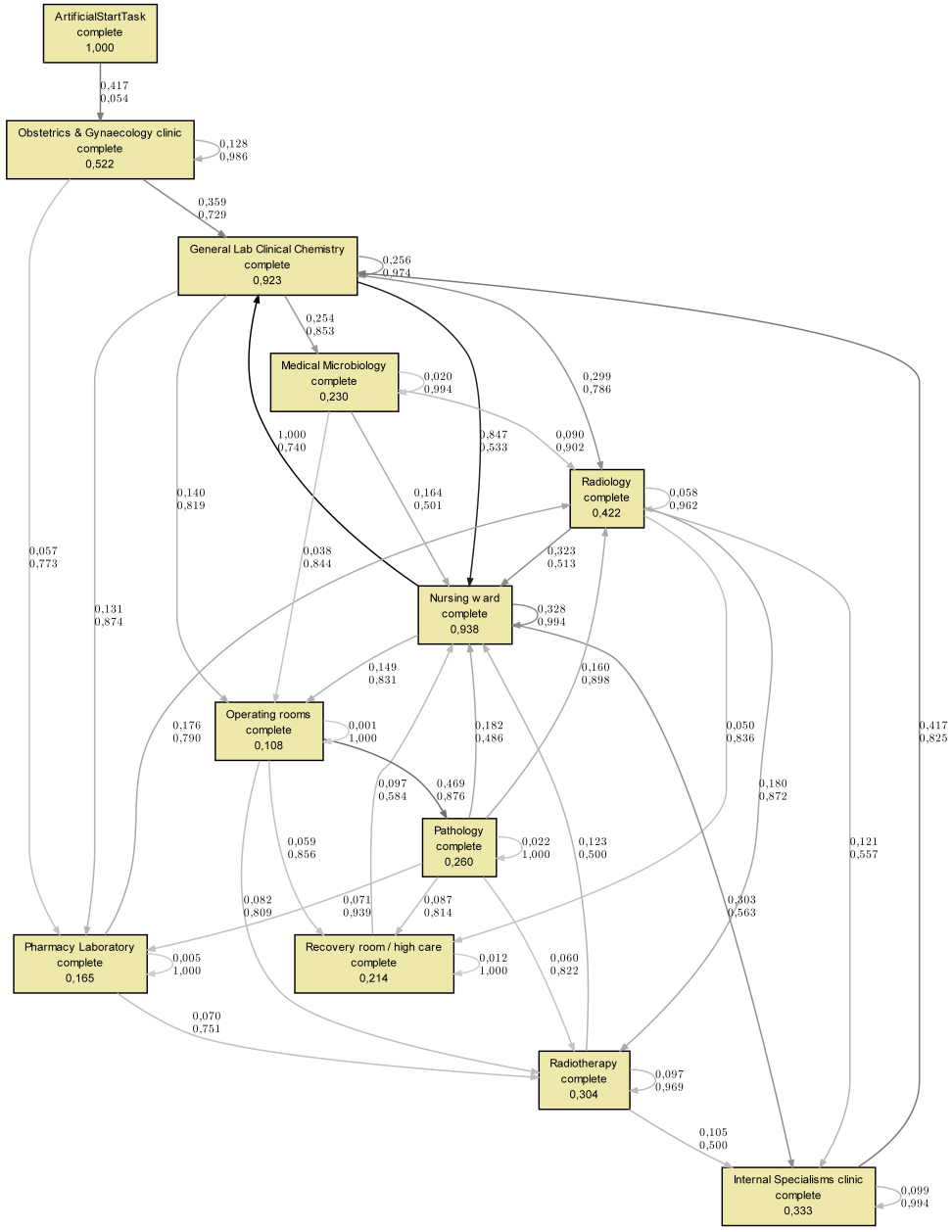


Figure 5.7: Fuzzy process model showing the relations between frequently occurring organizational units

straightforward since the granularity of the events is relatively coarse-grained. The data contains events such as *teletherapie - megavolt fotonen bestrali* and *brachytherapie - interstitieel - intensi*. Further, the radiotherapy department also carries out hyperthermia treatments, which are strictly speaking not radiotherapeutic, but often used in combination with radiation therapy. For chemotherapy, the selection of appropriate events was slightly more difficult due to the fact that this type of therapy is scattered between different organizational units. Nonetheless, we identified two important chemotherapeutic activities, viz. *paclitaxel* and *doxorubicine*. Finally, also surgical treatments should be taken into account. However, looking at the events pertaining to the Operating rooms department, there clearly exist two different types of procedures. On the one hand, the data set shows a multitude of diagnostic surgical procedures, whereas on the other hand only therapeutic operations are of interest given our current focus. Nonetheless the thin line between both, we were able to distinguish between the diagnostic or therapeutic nature of procedures by investigating the names of the events. For example, hysterectomies and vulvectomies were considered as therapeutic surgical activities, while hysteroscopies and urethrocystoscopies were not.

After singling out these therapeutic activities, 477 cases could be observed for which at least one therapeutic activity has taken place. The event log consisting of all these events was used to visualize the therapeutic activities by means of a process model. However, due to the large number of different surgical procedures, we renamed all these events to *surgery*, an abstraction which allows for more useful visualizations. The resulting process model as obtained with fuzzy mining is depicted in Figure 5.8. In contrast to Figure 5.7, the fuzzy net is slightly adapted by replacing the original figures in the nodes and on the arcs by more insightful statistics. As such, the nodes contain their frequency of occurrence, while the figure on each of the edges of the graph shows the number of times a case followed the transition from the source node to the target node.

The analysis allows for the formulation of the following findings:

- The fuzzy net shows a number of possible therapeutic choices. However, because the process model does not contain any such paths, it can be concluded that the combination of surgery or chemotherapy with radiotherapy occurs highly infrequent.

- The use of chemotherapy is rather limited despite the fact that regimens, i.e. combinations of different chemotherapy drugs, are often recommendable. In this way, we could only observe very few chemotherapeutic combinations of Paclitaxel with Doxorubicin. It should be further investigated why chemotherapy is underrepresented. One possible explanation might be that chemotherapeutic procedures might be carried out in peripheral hospitals, thus not captured in the data.

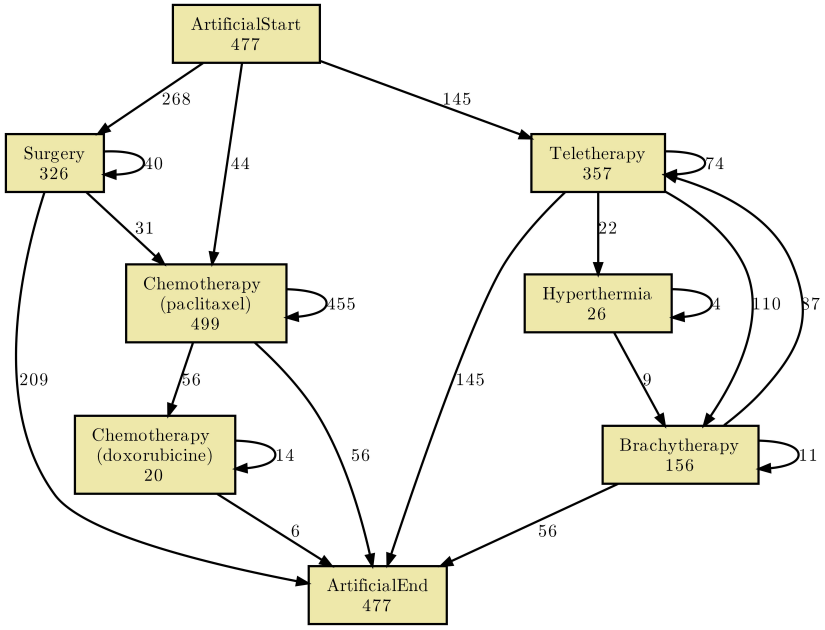


Figure 5.8: Adapted fuzzy process model showing the relations between different therapeutic activities

Visualizations using networked graphs. A second drill down approach consists of visualizing a subset of cases by means of a networked graph. This methodology is useful because it allows to visualize cases from different angles by supplementing control-flow information with other perspectives. The construction of a networked graph consists of three steps. First cases are selected based on some criterion. In this case, we considered all cervical cancer cases. Secondly, a Euclidean distance matrix is constructed denoting the distance between each pair of cases. This matrix is built by making use of the MRA (Maximum Repeat Alphabet) technique as proposed in [16]. The MRA technique relies on the identification of specific patterns which characterize the traces. Notwithstanding the fact that the authors employ the method for clustering log traces, we use the underlying distance matrix to construct a networked graph. Such a network graph connects nodes which represent a case in the data. For comprehensibility reasons, sparsification is applied in order to reduce the number of connections between the nodes because otherwise a fully connected graph is obtained. In this case, we applied K-nearest-neighbors with $K = 2$.

Figure 5.9 shows one visualization created with this methodology. Note that for visualization of the graph, we employed the Yifan Hu algorithm [75] as implemented in Gephi[4]. The graph shows all patients diagnosed with some type of cervical cancer. The distance between the nodes is determined by the MRA distance, which entails that nodes which are closer together present similar execution paths in terms of the therapeutic events they contain. The figure in the nodes denotes one representative case ID, with the size of the nodes representing the frequency of a certain sequence of therapeutic activities. Note that it was infeasible to represent all ID's in each of the nodes of the graph. Furthermore, the node colors indicate the application of some specific therapeutic procedure. As such, the green nodes denote the occurrence of chemotherapy. In contrast, the red nodes are cases where hyperthermia treatment is applied. From this visualization, it can be seen that a vast majority of cases do not rely on hyperthermia or chemotherapy. Cervical cancer is majorally treated by either surgery or radiotherapy (teletherapy/brachytherapy).

5.2.5 Conclusion

In this study, it was shown how intelligent analysis of clinical pathway data based on process mining techniques can deliver valuable insights into the actual carrying out of a care process. In a practical case consisting of data on the clinical pathways of 1.147 gynecological oncology patients at the AMC hospital, it was demonstrated how both drill up as well as drill down approaches are useful for care flow knowledge discovery. We are convinced that data analysis based on the

[4] www.gephi.org

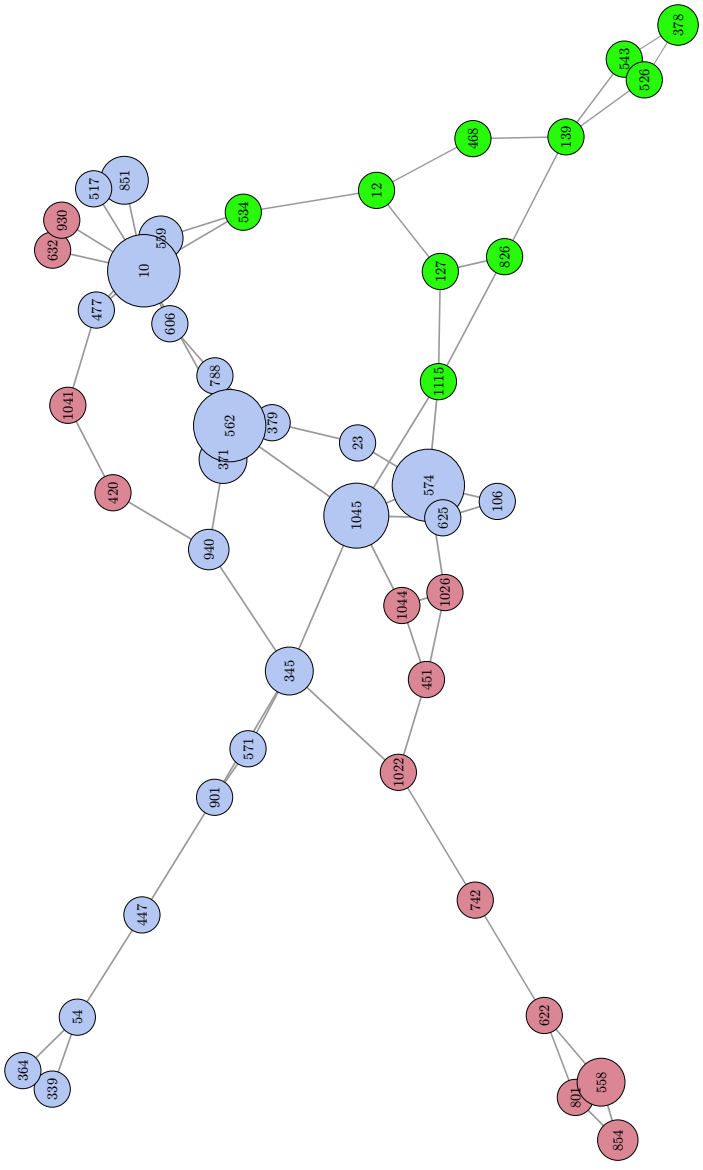


Figure 5.9: Networked graph visualizing cervical cancer cases

innovative techniques in the process mining domain is an ideal means for better streamlining and overall improvement of clinical care processes. In the future, we will focus on the development of novel methodologies for analyzing the complex data that is typically found in the logging infrastructures of healthcare information systems. As such, we will further elaborate the idea of networked graph visualizations and improve its integration with existing process discovery techniques. The major benefit of the technique is the enhancement of pure control-flow patterns with other data dimensions. Therefore, additional information, for instance on whether patients were cured or not, would instigate a wide variety of analysis possibilities.

5.3 Leveraging Process Discovery with Trace Clustering and Text Mining for Intelligent Analysis of Incident Management Processes

As explained in Chapter 4, solution strategies should be developed to cope with the problem that event logs display a too large variety in control-flow behavior so that the straightforward application of a single-model discovery technique will often result in an unsatisfactory result. Due to the low number of restrictions imposed by typical information systems in such flexible environments, ex post analysis of process execution data requires well-thought analysis methodologies. Trace clustering is one such technique which allows to reduce the complexity problem that single-model discovery techniques are confronted with. In this way, this case study presents how trace clustering can be used to separate execution traces into different groups for which a more accurate and comprehensible process model can be discovered. To this, a combination of text mining and decision tree learning is added to get insight into the atypical behavior in the system. In this way, traditional process discovery analysis is leveraged so as to discover more useful insights.

5.3.1 Problem Statement

Throughout this study, we will make use of a running example with the purpose to clarify the proposed approach. This running example is to be situated in the ICT-services department of the Catholic University of Leuven (KU Leuven). In this department, a dedicated team is responsible for the service desk whose main task is the resolution of IT-related problems. Hereto, a supporting information system is put in place, which registers the whole life cycle of incident management cases. As such, the data consists of the life cycle transitions of each of the cases as recorded by the logging infrastructure of the information system.

Incident management process: data description

Over a six month period, data about 2.726 cases was gathered. For each of these cases, two different types of data are available: case data and event log data. Case data, as depicted in Table 5.6, consists of an ID, a textual description and a type. In the system, four different types are distinguished: Bugfix/Repair, How-to/Advice, Productive Intervention and New Need. Note however that for 40% of the cases, the type is not specified. Next to type information, the data also consists of a textual description. The only restriction with regard to this field is that the text should be limited to at most 256 characters.

Table 5.6: Excerpt of case data

ID	Textual description	Type
Case 01	dump in professional expenses	Bugfix/Repair
Case 02	problem with assets	?
Case 03	alumni	How-to/Advice
Case 04	fw: certifications 1 - 0431	?
Case 05	standard check mark in field 'tax-x'	How-to/Advice
Case 06	accounting entry decentral part overhead	?
Case 07	failed wage booking	Productive Intervention
Case 08	failed booking 3/5/2010	Productive Intervention
Case 09	new employee invoices bureau	New need
Case 10	certifications a1-a431	How-to/Advice
Case 11	vat recovery 2009	Productive Intervention
...	...	...

Next to case data, there is also process-related information available in the form of an event log. This information consists of all the events registered for a certain case. These events correspond to the execution of an activity in the context of the life cycle of an incident management case. An excerpt of the data is shown in Table 5.7. Next to an ID, the event log consists of an originator who performed a certain activity, together with a timestamp that indicates the completion of this task. In the current data set, 15.552 events were recorded for the 2.726 cases. Overall, 73 different originators take part in the incident management process.

Process discovery in flexible environments

As stated earlier, process discovery is the most challenging learning task in the process mining domain. The crucial problem is caused by process data stemming from highly flexible environments. In such environments, little restrictions are put in place to limit the possible execution steps. In this way, there exists a wide variety of execution paths in the event log. Also for the running example data set, a wide variety of control-flow behavior is present, as illustrated by the fact that 338 different types of execution paths (further also denoted as grouped traces or *gt*'s) are found. Furthermore, these various types of traces differ significantly in terms of their frequency. This is depicted with the histogram in Figure 5.10, which has trace frequency intervals on the horizontal axis and the amount of distinct grouped traces belonging to such a trace frequency interval on the vertical axis. The figure clearly shows that the distribution of trace frequencies (i.e. the amount of cases that follow an identical solution trajectory in terms of the activities performed) is strongly skewed. On the one hand, a lot of low-frequent behavior is observed.

Table 5.7: Excerpt of event log data

ID	Originator	Activity	Timestamp
Case 01	System	Initialize	28/05/2010 11:07:30
Case 01	User 1A	Handle case	28/05/2010 11:14:21
Case 01	User 1A	Ask for feedback	28/05/2010 11:23:21
Case 01	System	On hold	28/05/2010 11:48:23
Case 02	User 2B	Handle case	28/05/2010 14:24:50
Case 02	User 2B	Solve with e-mail	28/05/2010 14:55:24
Case 03	System	Initialize	28/05/2010 17:37:49
Case 04	User 1A	Handle case	28/05/2010 17:50:54
Case 05	System	Initialize	31/05/2010 09:21:30
Case 03	User 1A	Handle case	31/05/2010 10:04:56
Case 03	User 1A	Ask for feedback	31/05/2010 10:10:59
Case 03	System	On hold	31/05/2010 10:24:41
Case 03	User 1A	SPAM	31/05/2010 11:54:05
Case 04	User 1A	Solve with e-mail	31/05/2010 12:21:56
Case 01	User 3C	Handle case	31/05/2010 14:26:05
Case 01	User 3C	Solve with e-mail	02/06/2010 09:15:00
Case 01	System	Receive solution feedback	02/06/2010 11:05:47
Case 01	User 3C	Accept solution	02/06/2010 11:22:35
Case 02	User 2B	Accept solution	08/06/2010 17:10:56
Case 04	User 1A	Accept solution	19/07/2010 17:34:30
...	...	...	...

For instance, there exist 211 cases that have a unique execution trace within the system. On the other hand, many cases comply with standard solution strategies in terms of their control-flow sequences. This is illustrated by the low number of highly frequent traces that occur in the event log. As such, more than 55% of all traces conform to one of the ten most frequent grouped traces.

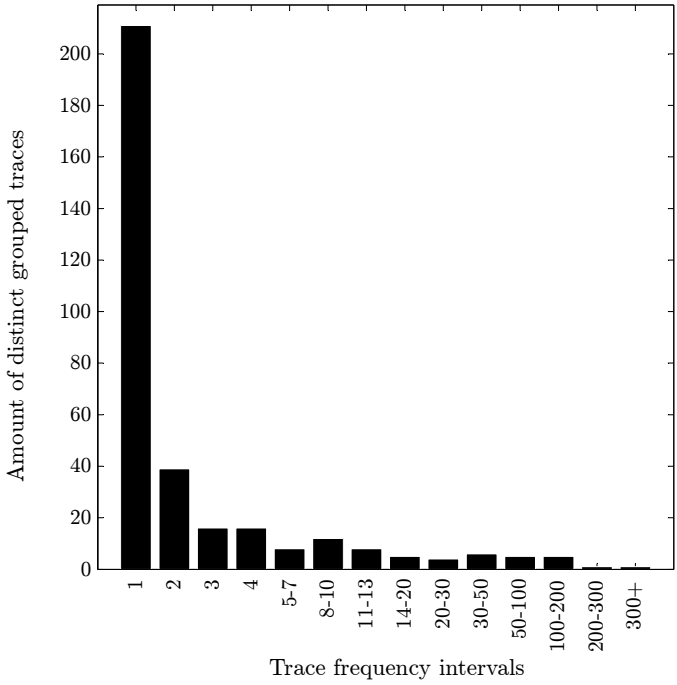


Figure 5.10: Histogram of trace frequency intervals for the event log

The availability of a wide range of execution behavior makes the application of single model process discovery techniques difficult. What is typically observed is that such techniques fail to discover an accurate and comprehensible process model out of the data. Also for the running example, this is the case. Figure 5.11 and Table 5.8 illustrate the problem at hand. The figure shows the Petri net process model as discovered by the HeuristicsMiner algorithm [153]. As can be seen from the table, this model scores badly with respect to accuracy. For instance, only 0,9% of the traces in the log can be correctly replayed (cfr. the number of successfully executed traces *set*). Similarly, behavioral recall (r_B^p) of this model equals 0,595 while behavioral precision (p_B) is 0,506. As shown in chapter 2, these values can be aggregated with the F-score to construct an overall measure for accuracy. The F1-score for this process model is only 0,547.

Next to accuracy, the comprehensibility of the discovered process model is

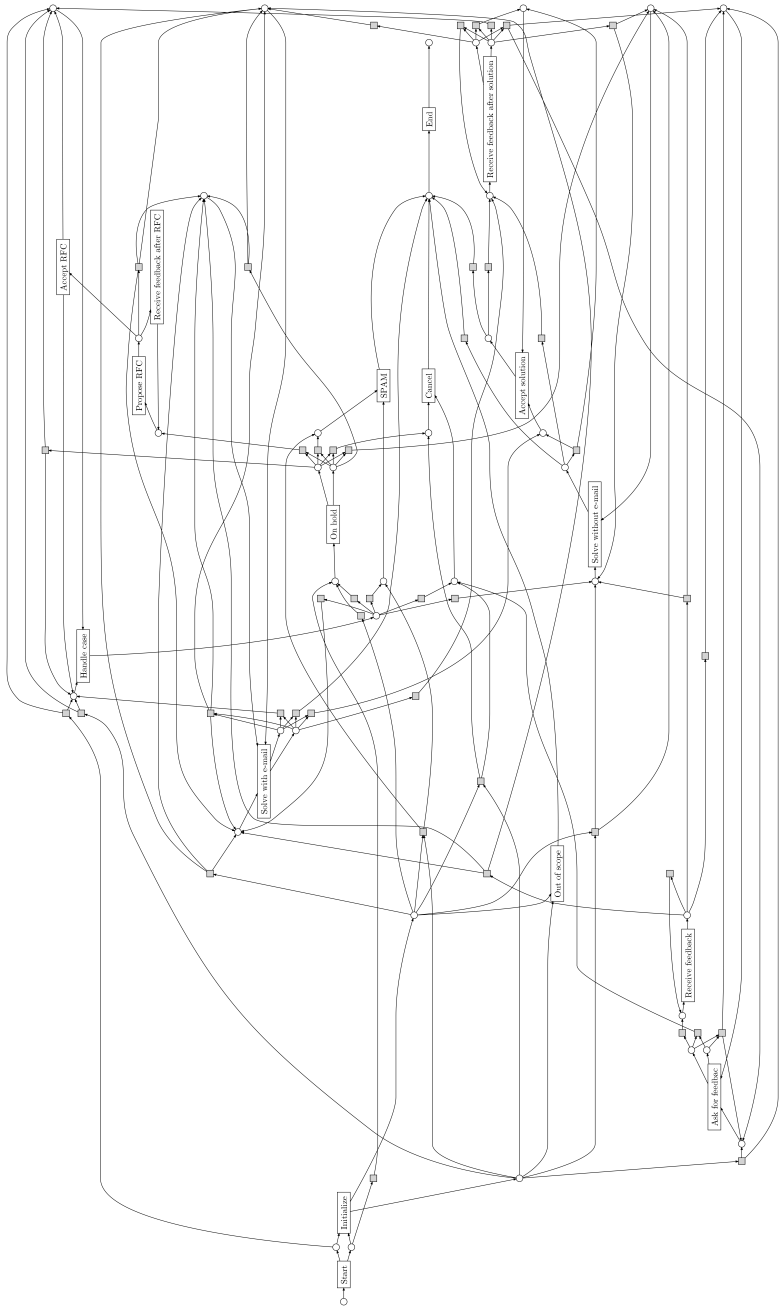


Figure 5.11: Petri net result of the HeuristicsMiner algorithm performed on the entire event log

mediocre as well. First of all, the discovered Petri net is an unsound workflow net. Furthermore, the model shows a high number of invisible transitions. Together with the high number of arcs connecting the nodes in the net, it should be clear that this model is not ideal for gaining insights into the actual behavior of this business process. Accordingly, we propose a complementary approach for analyzing the data that combines trace clustering and text mining, as shown in Figure 5.12.

		Complete event log
Event log	# traces	2.726
	# *gt*'s	338
	# events	15.552
Discovered process model	r_B^p	$0,595$
	p_B	$0,506$
	$F1_{r_B^p, p_B}$	$0,547$
	set	$0,009$
	# transitions	61
	# places	39
	# arcs	176
	sound	no

Table 5.8: Event log and discovered process model characteristics for the entire event log

Solution strategy

The solution strategy for leveraging process discovery in the current setting of a highly flexible incident management process consists of two important elements. First of all, trace clustering is applied with the purpose of dividing the event log traces into sub-event logs. This division will allow process discovery techniques to discover more accurate and comprehensible models from subsets of event log traces. With respect to trace clustering, the ActiTraC$^{\text{MRA}}$ as presented in Section employed. However, in contrast to the experimental evaluation of ActiTraC in Chapter 4, the option to separate out remaining behavior in a separate cluster is applied. These traces in the event log do not fit well in one of the created clusters because they exhibit atypical control-flow behavior. Therefore, these cases are separated out and form the subject of the text mining phase. In this second phase, a combination of text mining and data mining is proposed with the purpose of finding interesting patterns for the atypical cases. Instead of relying on control-

flow data (event log), we will make use of the case data, both the unstructured description field and the type information, since it can be expected that a lot of knowledge resides in these text fields. The whole solution strategy is represented in Figure 5.12.

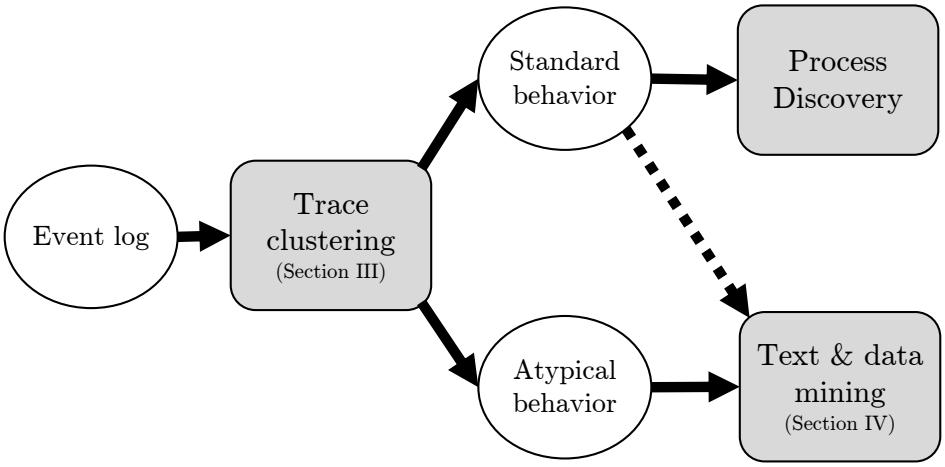


Figure 5.12: Overview of the proposed solution strategy

5.3.2 Trace Clustering

Within the field of process mining, Greco et al. [66] were the first to apply clustering techniques to event traces. The idea of grouping traces exhibiting common behavior is an intuitive approach to reduce the complexity of the discovery problem. By separating execution traces into different groups, one can rely on multiple process models to represent the variety in behavior of a certain event log. Also Song et al. [120] and Ferreira et al. [54] contributed to the development of trace clustering techniques. In this study, the ActiTraC$^{\text{MRA}}$ algorithm as outlined in Chapter 4 is used. Note that this technique is similar to the traditional MRA-based clustering technique proposed in [16]. However, instead of hierarchical clustering based on vectorized process instances, a semi-supervised clustering technique is put in place that closes the gap between clustering bias and evaluation bias. The capability of our algorithm to create a residual cluster with remaining, non-fitting traces for already created clusters is made use of. In this way, the identified atypical behavior is subject of analysis in Section 5.3.3.

ActiTraC$^{\text{MRA}}$ with a distinct cluster of remaining traces

In accordance with the solution strategy proposed in Section 5.3.1, ActiTraC$^{\text{MRA}}$ is applied to the event log of the running example. In this case we opted to create four clusters, with one of these the residual cluster. Furthermore, the frequency window was set to the 10% most frequent remaining traces (parameter α) and the minimum cluster size was set to 50% of the total number of remaining traces (parameter β). Tables 5.9 and 5.10 detail a comparison between the application of ActiTraC$^{\text{MRA}}$ and the standard MRA-technique with its optimal settings for this case (i.e. using F-score similarity). The tables show both event log information with respect to the clusters created as well as accuracy and comprehensibility metrics for the underlying process models.

		Clus_1	Clus_2	Clus_3	Clus_resid
Event log	# traces (%)	1.372 (50%)	698 (26%)	225 (8%)	431 (16%)
	# *gt*'s (%)	27 (8%)	27 (8%)	24 (7%)	260 (77%)
	# events	6.612	3.920	1.582	3.438
Discovered process model	r_B^p	1,000	1,000	1,000	0,491
	p_B	0,996	0,975	0,891	0,199
	$F_{r_B^p, p_B}$	0,998	0,987	0,943	0,284
	set	1,000	1,000	1,000	0,000
	# transitions	22	24	23	52
	# places	11	14	13	34
	# arcs	44	48	46	146
	sound	yes	yes	yes	no

Table 5.9: Event log and discovered process model characteristics for ActiTraC$^{\text{MRA}}$ clustering

The results show a clear difference between both approaches. While ActiTraC$^{\text{MRA}}$ presents three trace clusters with highly accurate underlying process models and one residual cluster with low accuracy, the standard MRA-technique discerns only one cluster (clus_3) with an acceptable accuracy level. If one averages the F-scores of the process models according to their trace frequencies, it can be found that ActiTraC$^{\text{MRA}}$ realizes a weighted average F-score of 0,88 while the original MRA-technique equals 0,69. As such, from an accuracy perspective, our novel trace clustering technique can be considered better performing.

From an event log perspective, the optimization strategy of ActiTraC$^{\text{MRA}}$ drives the algorithm towards the creation of fairly large clusters in absolute terms, hereby clustering only a limited number of grouped traces together. The residual cluster (clus_resid) contains about 16% of the total number of traces, while this cluster contains more than 75% of the grouped traces. The distribution of

		Clus_1	Clus_2	Clus_3	Clus_4
Event log	# traces (%)	57 (2%)	356 (13%)	1146 (42%)	1167 (43%)
	# *gt*'s (%)	40 (12%)	106 (31%)	44 (13%)	148 (44%)
	# events	591	2.843	5.129	6.989
Discovered process model	r_B^p	0,702	0,701	0,979	0,809
	p_B	0,304	0,313	0,891	0,434
	$F_{r_B^p,p_B}$	0,424	0,433	0,922	0,565
	set	0,000	0,014	0,931	0,198
	# transitions	26	34	30	37
	# places	18	21	14	22
	# arcs	63	80	60	83
	sound	no	no	yes	no

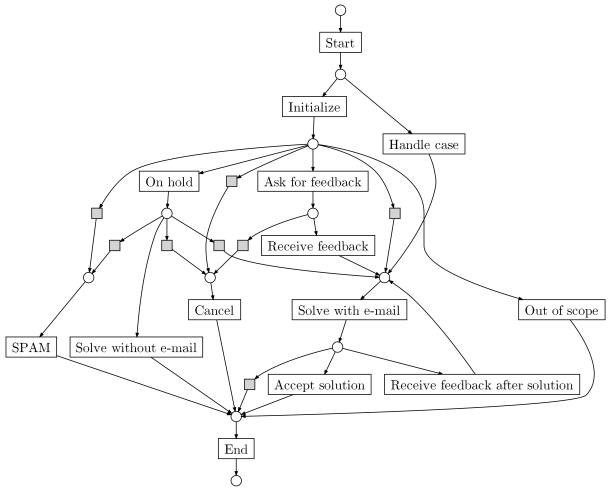
Table 5.10: Event log and discovered process model characteristics for standard MRA clustering

grouped traces over the different clusters is slightly more balanced for the standard MRA-technique. A final element of comparison involves scalability. Since the computational complexity of ActiTraC$^{\text{MRA}}$ is significantly larger than standard MRA, it is logical that the run times of the algorithms differ. In this case, ActiTraC$^{\text{MRA}}$ takes on average about 33 seconds to yield a result, while standard MRA computes the result in about 2 seconds on a standard stand-alone PC. Note that an exhaustive analysis of the scalability of ActiTraC$^{\text{MRA}}$ is out of the scope this study.

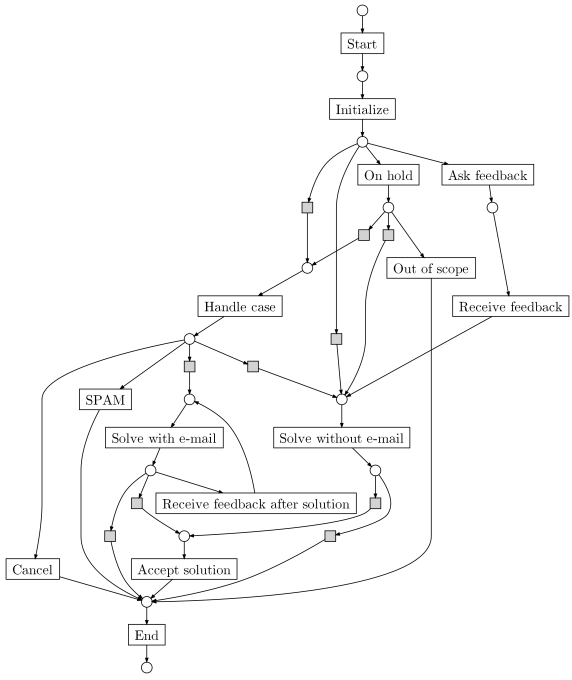
The resulting process models as discovered from the ActiTraC$^{\text{MRA}}$ clustering result with the HeuristicsMiner algorithm are shown in Figure 5.13. Note that the process model for the residual cluster is not represented for two reasons. First of all, the according process model is inaccurate and incomprehensible. Secondly, these residual traces will be the subject of a different knowledge discovery approach as described in the next section.

5.3.3 Analysis of Atypical Behavior

As described earlier, ActiTraC$^{\text{MRA}}$ results in four trace clusters of which three clusters present accurate and comprehensible process models. However, the fourth cluster contains all the residual traces that do not fit one of the process models underlying these three clusters. As such, these traces are considered as atypical control-flow behavior. It is important to note that the non-residual clusters also contain low-frequent behavior, so not all low-frequent behavior is to be considered as atypical. In this section, it is outlined how a combination of text mining and regular data mining can be made use of in order to discover interesting patterns for this atypical behavior.

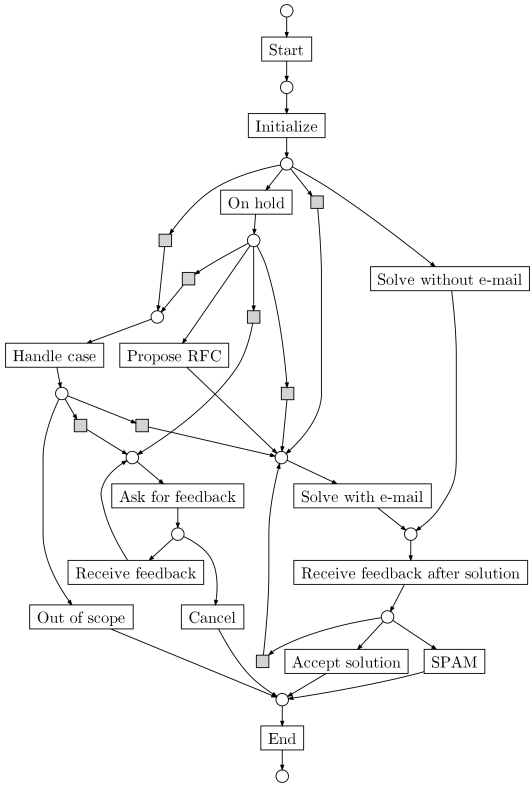


(a) Discovered Petri net for clus_1



(b) Discovered Petri net for clus_2

Figure 5.13: Petri net result of the HeuristicsMiner algorithm performed on the trace clusters discovered by ActiTraC$^{\text{MRA}}$



(c) Discovered Petri net for clus_3

Figure 5.13: Petri net result of the HeuristicsMiner algorithm performed on the trace clusters discovered by ActiTraC$^{\mathrm{MRA}}$

Characterizing deviating traces

The key idea behind the solution strategy is leveraging the knowledge hidden in the case data of each of the execution traces. As such, this phase consists of the application of text mining and decision tree learning with the purpose of finding patterns that characterize peculiar control-flow behavior. This behavior is not common to a majority of the cases in the information system and therefore separated out by ActiTraC$^{\text{MRA}}$ in a residual cluster. We would like to stress that we do not consider these traces as incompliant. Notwithstanding the fact that there might exist cases that violate certain business rules or constraints, we assume that the large majority of this atypical control-flow behavior is allowable. Accordingly, it seems interesting to look at other characteristics than the typical control-flow aspects of these cases. In this way, making use of the additional case information becomes an interesting approach.

Text mining

For the running example, case data is limited to type information and a fully unstructured textual description. Accordingly, it is this textual description field that is subject of a text mining analysis. The purpose of text mining is the extraction of frequent keywords that subsequently can be used in a classification learning setting which distinguishes between typical and atypical behavior. Text mining from these unstructured text data requires the filtering of stop words and stemming to prove effective. These steps were performed by making use of the RapidMiner[5] software. Since textual descriptions are available in both English and Dutch, stop word filtering and stemming were performed for both languages. Table 5.11 lists the most frequent words in the data set. Since the majority of textual descriptions are written in Dutch, the most frequent words are in Dutch as well. Note that there is no optimization procedure put in place that matches different words with the same meaning in Dutch and English into one attribute word. The type information together with the tokenized textual descriptions are now available as attributes for a standard classification learning task.

Classification results

As stated earlier, text mining is combined with decision tree learning in order to discover useful patterns from case data. To do so, the C4.5-technique [103] was applied. Note that the data set consists of both typical traces that belong to one of the non-residual clusters and atypical behavior. Since the data set is strongly skewed as there exists much more typical traces as compared to atypical, we applied random undersampling in order to balance the data. As such, the

[5] http://rapid-i.com

Word (English translation)	Frequency
job (task)	410
step (step)	254
lutick (lutick)	129
fout (error)	128
probleem (problem)	109
loket (counter)	101
foutieve (erroneous)	87
bim (bim)	84
kredietkaart (credit card)	80
isp (isp)	78
nieuw (new)	77
beroepskost (professional expense)	72
factuur (invoice)	64
aanwezig (present)	57
student (student)	57
sap (sap)	54
...	...

Table 5.11: Frequent word list

input data for the tree learning algorithm consists of 862 traces. Furthermore, we applied 10-fold cross-validation for an assessment of the classification accuracy of C4.5 decision trees.

The result of this data mining analysis is represented in Figure 5.14. The tree classifies typical (0) and atypical traces (1) by making use of the type information and the tokenized textual description. It is recognized that the accuracy level of 63% is moderate, however, this is mainly due to the fact that the logged textual descriptions are fairly short. The methodology could be more advantageously applied in case there would be more textual information available. However, the logging infrastructure currently could not provide more elaborate case information.

With respect to the patterns discovered by C4.5, it can be observed for instance that Bugfix/Repair cases typically deviate from standard control-flow behavior. Furthermore, we observe that cases where the word "problem" appears in the text field but without "sap", also differ from typical control-flows. Based on these insights, the ICTS services department might undertake an investigation of these cases in order to find out the exact reasons of deviation. Note that the proposed methodology would benefit strongly from an increase in the amount of case data registered by the logging infrastructure. Both structured as well as unstructured

data fields might increase the potential of accurate discovery of patterns that provide insight into atypical process behavior.

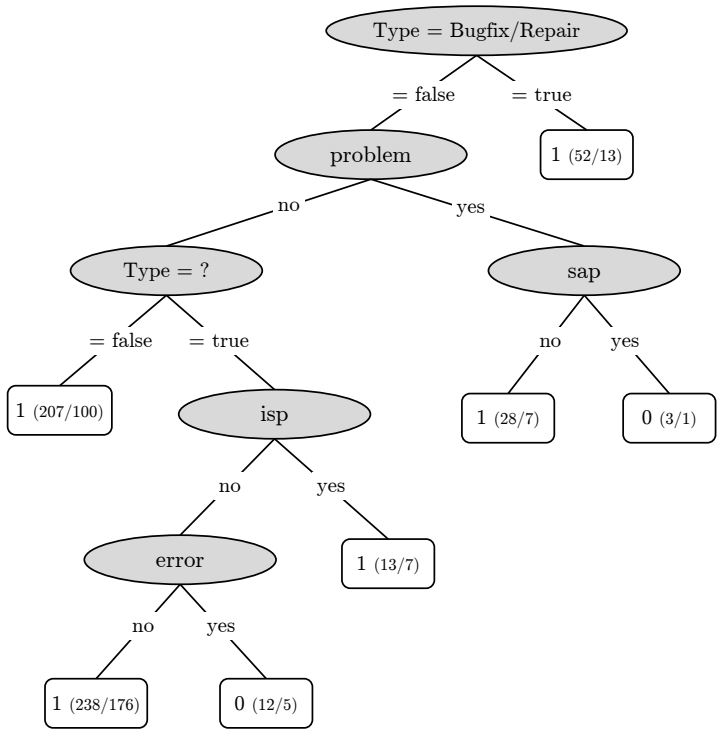


Figure 5.14: C4.5 decision tree with 63% accuracy (10-fold cross-validation)

5.3.4 Discussion

The discussion section focusses on both elements of the proposed solution strategy.

Trace clustering

The newly proposed trace clustering technique ActiTraC$^{\text{MRA}}$ shows some interesting results. By optimizing the underlying process models of the clusters, it is demonstrated that currently available trace clustering techniques can be improved. However, the semi-supervised learning strategy entails a definite increase in computational complexity. In future research, it should be investigated how the scalability of the proposed technique behaves in other real-life environments. Also a comparative evaluation with other available trace clustering techniques is subject of future research. Furthermore, the parameters of the algorithm should

be further investigated as well as finding an optimal way for determining the right amount of clusters that should be created.

Originator effect

A second element of discussion considers the setup of the text and data mining phase. In the running example, we made use of all available data except for originator information. However, we tested the addition of originator information in the second data mining phase. Intuitively, it can be expected that differences in working methods between originators are an important source of atypical process behavior. Nonetheless, the results of the experiment indicated that originator information has very limited classification power with respect to distinguishing between typical and atypical process behavior. This indicates that, for the case at hand, root causes of atypical process behavior are not strongly related to differences between originators.

5.3.5 Related Work

As explained earlier, process discovery [10, 63, 118, 131, 153] and trace clustering [15, 16, 54, 66, 120] are important foundations of this work. Nevertheless, also conformance checking techniques are key building blocks. For instance, the ICS-fitness metric [152] is the underlying fitness metric of ActiTraC$^{\text{MRA}}$. For evaluation, we employed the recall and precision metrics as described in the second chapter. Other conformance checking metrics could have been put in place. For instance, the original conformance metrics proposed in [112] are a viable alternative. With respect to precision, the ETC-metric in [96] is very similar to behavioral precision (p_B) as used in this study.

Further, the application of rule learning and decision tree algorithms is not new in the process mining domain. For instance, Rozinat et al. [113] developed a technique that investigates the decision points in a process model based on case data. Furthermore, Maruster et al. [91] proposed the use of Ripper [30] to the process discovery learning problem. Also Linear Temporal Logic (LTL) is often applied in the field of process mining [137]. For instance, Maggi et al. [87] founded their declarative process discovery technique on LTL.

5.3.6 Conclusion

In this study, we have developed a novel solution strategy for leveraging process discovery techniques in highly flexible environments where traditional single-model discovery techniques underperform. Our approach consists of the application of trace clustering and a combination of text mining and data mining in order to extract useful knowledge from the original data. With respect to clustering, we

proposed an adaptation of the existing MRA-technique in the sense that we altered the underlying clustering technique from simple hierarchical clustering to a semi-supervised learning algorithm that greedily optimizes the accuracy of the underlying process models. The results in the context of an incident management process show that the proposed methodology is feasible and valuable. Thanks to the novel trace clustering technique, more accurate and comprehensible process models were discovered that represent the typical process behavior in the system. Furthermore, text mining and decision tree learning provided useful results with respect to characterizing the atypical behavior relying on case data instead of control-flow information.

In future work, we plan to further develop the novel trace clustering technique. Furthermore, we think that combining process discovery, text mining and data mining should be the subject of increased research efforts. Accordingly, we intend to improve the proposed approach, for instance in the context of healthcare processes where the level of flexibility is even higher.

Chapter 6

Conclusion

In the last decade, Process Mining has continuously broadened its prominence as the most important discipline providing a comprehensive set of techniques for the analysis of business processes. As explained, Process Mining bridges the two important research domains of Business Process Management (BPM) and Knowledge Discovery in Databases (KDD). With event logs being the cornerstones of analysis, process mining techniques possess the capabilities to gain insight into the actual way of working in the context of a certain business process. Most importantly, due to their reliance on execution data, the objectivity of these techniques is to be considered as their unique selling proposition. Despite the hard work of many researchers around the world, the widespread application of process mining techniques in industry still remains behind. Notwithstanding the level of academic maturity of the field, it cannot be denied that there is more work to do towards effectively crossing the chasm.

This dissertation covers multiple areas of the process mining domain. Chapter 2 elaborates on process discovery evaluation metrics with the main research objective being the conception of a comprehensive evaluation framework for discovered process models. The contributions of this chapter are further employed for a multi-dimensional quality assessment of currently available process discovery techniques in Chapter 3. Furthermore, a novel technique to cluster event traces is proposed in Chapter 4. The ActiTraC algorithm targets to bridge the gap between clustering bias and evaluation bias which is a key difficulty for a large majority of existing trace clustering techniques. Finally, the findings of this dissertation were further validated with three case studies. To conclude this thesis, the main research contributions are revisited. Moreover, multiple challenges with respect to different topics that were covered in this thesis remain to be addressed. Accordingly, a couple of issues for future research are identified.

6.1 Thesis Contributions

In this dissertation, we have described contributions in three different areas of the process mining research field.

Evaluating discovered process models. In Chapter 2, an assessment is provided of currently available process discovery evaluation metrics. This assessment revealed that many of these metrics suffer from multiple drawbacks. In essence, the process mining field still lacks an implemented, comprehensive framework for the evaluation of discovered process models. Despite the fact that evaluation dimensions such as recall, precision and generalization have been pinpointed from a theoretical point of view, the actual implementation in terms of metrics still requires further research. This thesis describes the application of the F-score or F-measure for process discovery evaluation. The F-score addresses one other question related to the absence of a comprehensive evaluation framework, namely the unavailability of methodologies that specify how to combine and prioritize different metrics to obtain an overall assessment of process model accuracy. This is an important question because discovered process models are likely to differ with respect to multiple dimensions at once. Relying on a technique to induce artificial negative events into an event log, the proposed F-score combines recall and precision, allowing to weigh these dimensions differently. Note that the capability to prioritize an evaluation dimension in process discovery is a desirable characteristic of a process discovery evaluation framework. This is because recall is the predominant evaluation dimension for discovered process models because the main objective of a process discovery technique is providing a sound reflection of the behavior in the log. Accordingly, for both researchers as well as practitioners the availability of a straightforward method to prioritize recall over any other evaluation dimension is convenient. Furthermore, the assessment of other evaluation dimensions is often influenced by the level of recall of a discovered process model. For instance, existing precision metrics relying on trace replay depend on whether traces fit the model. This is because in case they don't, one is always faced with a complex choice to determine the best option to mimic the non-fitting trace in the discovered process model. Despite more advanced model-log alignment techniques such as described in [3], making a replay choice for non-fitting traces will always influence a replay-based precision metric. In sum, the F-score evaluation approach is considered a valuable addition towards the development of an evaluation framework for discovered process models.

Benchmarking process discovery techniques. The next contribution of this dissertation consists of the first large-scale benchmarking study of process discovery techniques using real-life event logs. Assessing the quality of process discovery

techniques is an essential element for both process mining research as well as for the use of process mining in practice. Up till now, evaluation of process discovery techniques was performed in very diverse ways, but mostly on artificial data sets. In Chapter 3, a multi-dimensional evaluation study was carried out comparing the accuracy, comprehensibility and scalability of state-of-the-art process discovery techniques on eight real-life event logs. In general, HeuristicsMiner was found the most appropriate technique in a real-life context. Furthermore, the F-score methodology was demonstrated to be a practical approach towards determining the overall accuracy performance of a process discovery technique. Nevertheless, the study also unveiled a couple of challenges. For instance, proficient quantification of the comprehensibility of discovered process models remains difficult. Although the use of complexity metrics from the process modeling domain was found to be applicable, the subtle differences between mined and modeled process models bring about the need for tailor-made solutions. Finally, the problem of representational bias in process mining was also identified as an important shortcoming of currently available process discovery techniques. In the benchmarking study, it was shown that single model Petri net discovery techniques often fail to simultaneously address the challenge of accurate, comprehensible and fast model discovery. One possible strategy to further improve the results of process discovery techniques is a reduction of the search space or a redefinition of the modeling language so as to accommodate for the specificities of process discovery and process discovery evaluation.

Active trace clustering. The third important contribution consists of a new approach for clustering log traces. Trace clustering is one possible strategy to resolve the problem that currently available process discovery algorithms are unable to discover accurate and comprehensible process models out of event logs stemming from highly flexible environments. The main advantage of trace clustering is that an event log can be split up so that process models can be discovered for multiple subsets of traces. The new algorithm, called ActiTraC, has its foundations in the observation that existing trace clustering algorithms suffer from a severe divergence between the clustering bias and the evaluation bias. This problem is resolved with an active learning inspired approach that centers on optimizing the combined process model accuracy. In experimental evaluations, ActiTraC was shown to significantly improve the accuracy and complexity of the process models underlying the discovered trace clusters as compared to existing trace clustering techniques. Furthermore, ActiTraC is a flexible algorithm that can be adapted for instance towards discovering more domain-relevant clusters. This is because it provides the option to include a distance matrix between log traces so that the grouping of more similar traces is enforced. Note that such a distance matrix can be constructed by using different types of information. In this dissertation, only

control-flow characteristics of the traces were taken into account. However, other types of information can be highly useful as well in order to steer the ActiTraC algorithm.

Case studies. Finally, Chapter 6 describes three case studies that further validate the outlined contributions. Because these studies are situated in the financial industry, healthcare, and public services respectively, they reflect the wide applicability of process mining techniques.

6.2 Issues for Future Research

The last section of this concluding chapter outlines a number of future research issues in the areas of process discovery evaluation and trace clustering.

6.2.1 Process Discovery Evaluation

As outlined earlier, a comprehensive evaluation framework for process discovery techniques is still not fully realized. In our opinion, this is due to four important challenges.

Computational complexity of trace replay in Petri nets. The calculation of evaluation metrics most often relies on trace replay in a Petri net. In case such a model contains invisible or duplicate transitions, traditional trace replay techniques [107] require heuristics in order to keep their computational complexity under control. However, these heuristics often interfere with the exact calculation of evaluation metrics. Recent works by Adriansyah et al. [1, 2, 3, 4] have proposed solutions for dealing with complexities such as invisible and duplicate transitions by *aligning* an event log and a process model. Hereto, the A* algorithm, a well-known algorithm for shortest path calculation in a directed graph [45], is employed. The methods proposed by Adriansyah et al. are capable of dealing with several problematic constructs such as invisible and duplicate transitions from which for earlier trace replay techniques suffer. Yet, as illustrated in [1], these very recent methods are sometimes confronted with high run times for complex process models as well. Another approach for alleviating the computational complexity of the calculation of conformance metrics is proposed by Weidlich et al. [151]. Instead of replaying an event log in the model, they compare a model and a log based on their respective behavioral profiles. An advantage of this technique is that time-consuming state-space calculations can be avoided. However, the *global* perspective often leads to underestimations of the measured conformance dimension, for instance when loops occur. As such, the complex trade-off between

accurate estimation of the evaluation metric and computational complexity of the calculation method remains an issue for future research.

Punishing overfitting. A second problem is the lack of a good quantifier of the generalization dimension. The main issue concerning the definition of a generalization metric lies in the fact that one has to reason about the *likeliness* of unseen behavior. It is a definite challenge to design methodologies that can foresee which kind of unseen behavior should be accepted by the model. However, without quantification of the generalization capability of a process model, it is very difficult to punish overfitting process models.

Combining and prioritizing evaluation dimensions. Notwithstanding that this dissertation proposes a method to combine and prioritize two important evaluation dimensions, different questions remain with respect to the integration of the identified evaluation dimensions. One can think of an extension of the F-score towards the three accuracy evaluation dimensions by calculating a weighted harmonic mean of recall, precision and generalization scores. Note that a robust quantifier of generalization is required hereto. However, a next issue is the combination of accuracy and complexity. In Chapter 2, accuracy and comprehensibility were identified as the two top-level evaluation dimensions. However, in the process mining literature, there has been little evidence on how these two more high level evaluation dimensions should be aggregated. In [22], Buijs et al. propose to consider *simplicity* as a fourth evaluation dimension next to recall, precision and generalization. As such, they propose to construct a user-defined weighted average of these four dimensions so as to steer their Evolutionary Tree Miner (ETM) algorithm. Notwithstanding the fact that it is interesting to observe that multiple dimensions can be aggregated in a straightforward way, the question remains open as to how one can prioritize between different dimensions without any prior knowledge. Furthermore, this approach has to be validated in more difficult settings so as to verify its robustness. Also, the capturing of simplicity in a single variable based on a translation to process trees [21] seems rather simple as it can be expected that other characteristics of the process model might influence the true complexity. Therefore, in our opinion, the combination of the general dimensions of accuracy and comprehensibility for process discovery evaluation needs to be explored further.

It is pointed out that in traditional statistics, methodologies to combine model accuracy and model complexity are available. For instance, the Akaike Information Criterion (AIC) [6] and the Bayesian/Schwarz Information Criterion (BIC) [116] are robust model selection criteria that aggregate the likelihood of a model and the size of the model in terms of the number of free parameters that should be estimated. However, the application of the idea behind such statistical methods

is not straightforward for process discovery evaluation. One of the main issues is that the *size* of a discovered process model is not always the best quantifier of process model complexity.

The completeness assumption. Event logs are rarely complete which complicates the definition of precision and generalization metrics. As such, the precision metric outlined in Chapter 2 as well as the ETC precision metric defined by Muñoz-Gama and Carmona [96] suffer from similar, overly strict completeness assumptions. Developing evaluation metrics that take into account the natural lack of completeness of event logs requires two key elements. First of all, currently available precision metrics and eventual generalization metrics should be adapted so that their respective completeness assumptions can be relaxed. For instance, with respect to artificial negative event based precision, there exists opportunities to relax the strict completeness assumption by an intelligent use of prefix windows. Secondly, such methods to relax the completeness assumption should be combined with statistical methods to assess the estimated completeness of an event log. By assessing the severance of incompleteness, one can for instance define confidence intervals for precision and generalization metrics. Note that in [156] and [157], statistical techniques are shown to be useful to estimate the completeness of an event log.

6.2.2 Trace Clustering

The main research challenges for the development and further improvement of trace clustering techniques are caused by the fact that trace clustering is expected to deal with multiple objectives. First of all, trace clustering aims at creating more accurate and more comprehensible process models for subsets of traces. Next to these two expectations, trace clustering techniques are expected to group similar, domain-relevant behavior. Accordingly, trace clustering techniques are faced with a multiobjective optimization problem.

Domain-driven trace clustering and the curse of dimensionality

A large majority of existing trace clustering techniques target the creation of domain-relevant clusters in the first place. In order to realize this objective without any additional information beyond control-flow information, most of these techniques rely on a vector-based approach where the characteristics of traces are translated into an attribute-value setting. Based on these vectors, distances can be calculated between traces so as to apply traditional clustering techniques from the data mining domain. However, due to the fact that control-flow characteristics can be expressed by a huge amount of attributes, existing trace clustering techniques

suffer heavily from the *curse of dimensionality*. As detailed in [74], as soon as the data dimensionality is high and the number of relevant dimensions is rather low, the curse of dimensionality hampers any analysis task. Due to the fact that it can be expected that a limited set of features determines the domain-relevancy for a set of traces, solutions for dealing with this dimensionality problem are required. Thus far, this has not been investigated within the process mining domain. In the domain of data mining, the problem of the curse of dimensionality in unsupervised settings has also received only modest attention. However, techniques such as shared nearest-neighbor (SNN) [74] and feature subset selection methods as described in [50] can be expected to improve currently available, vector-based trace clustering techniques.

Integrating domain- and accuracy-driven trace clustering

In spite of the focus of existing trace clustering techniques on the creation of domain-relevant clusters, it is argued in this dissertation that the accuracy of the process models underlying the trace clusters can be of interest as well for steering a trace clustering algorithm. More specifically, the ActiTraC algorithm was designed so as to incorporate the optimization of the accuracy of the underlying process models in the clustering methodology by means of a semi-supervised learning approach. Furthermore, the implemented selective sampling strategy allows for the incorporation of any kind of distance matrix between traces in order to combine the need for accurate and domain-relevant trace clusters. However, the incorporation of the MRA-based Euclidean distance between traces did not significantly improve the creation of meaningful clusters in the supervised experiment. Therefore, there exist opportunities for future research to further improve the combination of both the need for domain-relevant trace clusters as well as accurate underlying process models.

Comprehensibility-driven trace clustering

A final element with respect to trace clustering that has not been addressed in this thesis concerns the problem of model complexity. Typically, understandability of the process models underlying the created trace clusters is a key determinant of the usefulness of trace clustering, as is the case with the requirements of accuracy and domain-relevancy. As such, the design of the *ideal* trace clustering algorithm should involve a strategy that allows to optimize three important factors at the same time. It is considered future work to investigate the possibilities of including complexity metrics into a semi-supervised clustering technique so as to steer the algorithm towards even more useful results. It should be noted that this research problem partially overlaps with research to be conducted in the area of process

discovery evaluation on how to combine process model accuracy and process model complexity.

Appendix A

Detailed Accuracy Results for the Real-Life Event Logs

In this appendix, the detailed results are provided of the experiments as part of the benchmarking study described in Chapter 3. The tables outline multiple recall and precision metrics for the different data sets as well as the run time of each of the algorithms.

	Recall					Precision				Run time
	f	r_B^p	$PF_{Complete}$	PM	set	$a_B^{'}$	s_B^n	p_B	etc_P	
AGNEsMiner	0,426	0,895	na	na	0,673	0,522	0,674	0,097	na	2d:23h:32m:54s
α^+	0,703	0,804	na	na	0,084	0,694	0,077	0,033	0,188	1s
α^{++}	0,663	0,406	na	na	0,000	0,643	0,490	0,030	1,000	1s
DT Genetic Miner	0,823	0,699	na	na	0,553	1,000	0,740	0,095	na	4d:13h:49m:15s
Flower	1,000	1,000	0,145	0,000	1,000	0,694	0,000	0,038	1,000	1s
Genetic Miner	0,121	0,522	0,599	0,030	0,067	0,726	0,966	0,372	1,000	22d:11h:58m:19s
HeuristicsMiner	0,203	0,400	0,000	0,000	0,000	0,735	0,931	0,184	na	5s
ILP Miner	1,000	1,000	na	na	1,000	0,564	0,407	0,062	0,541	5m:36s

Table A.1: Accuracy metrics for UFM data set

	Recall					Precision				Run time
	f	r_B^p	$PF_{Complete}$	PM	set	$a_B^{'}$	s_B^n	p_B	etc_P	
AGNEsMiner	0,893	0,980	na	na	0,000	1,000	0,965	0,624	0,706	23m:32s
α^+	0,832	0,729	na	na	0,000	0,326	0,801	0,181	0,158	1s
α^{++}	na	na	na	na	na	na	na	na	na	1s
DT Genetic Miner	0,982	0,982	0,000	0,000	0,970	0,594	0,948	0,533	na	1d:6h:2m:26s
Flower	1,000	1,000	0,571	0,000	1,000	0,420	0,000	0,057	0,125	1s
Genetic Miner	0,753	0,879	0,652	0,000	0,110	0,742	0,980	0,724	na	1d:8h:4m:49s
HeuristicsMiner	0,996	0,875	0,662	0,000	0,973	0,856	0,987	0,803	0,550	4s
ILP Miner	0,899	0,990	na	na	1,000	0,493	0,770	0,205	0,404	29s

Table A.2: Accuracy metrics for P2P data set

	Recall					Precision				Run time
	f	r_B^p	$PF_{Complete}$	PM	set	$a_B^{'}$	s_B^n	p_B	etc_P	
AGNEsMiner	0,731	0,853	na	na	0,451	0,570	0,905	0,397	na	47m:56s
α^{+}	0,527	0,367	0,000	0,000	0,000	0,499	0,991	0,741	1,000	1s
α^{++}	0,580	0,428	0,000	0,000	0,000	0,470	0,985	0,674	1,000	1s
DT Genetic Miner	0,573	0,926	na	na	0,028	0,685	0,859	0,325	na	10h:22m:32s
Flower	1,000	1,000	0,288	0,000	1,000	0,520	0,000	0,068	0,208	1s
Genetic Miner	0,439	0,912	0,906	0,531	0,070	0,680	0,960	0,625	na	4h:23m:57s
HeuristicsMiner	0,485	0,703	0,554	0,010	0,049	0,868	0,944	0,481	na	4s
ILP Miner	0,933	1,000	na	na	1,000	0,749	0,513	0,131	0,371	26s

Table A.3: Accuracy metrics for KHD data set

	Recall					Precision				Run time
	f	r_B^p	$PF_{Complete}$	PM	set	$a_B^{'}$	s_B^n	p_B	etc_P	
AGNEsMiner	0,758	0,847	na	na	0,065	0,675	0,891	0,285	1,000	50m:40s
α^{+}	0,586	0,336	na	na	0,000	0,438	0,819	0,088	1,000	1s
α^{++}	0,694	0,309	na	na	0,000	0,500	0,881	0,118	1,000	1s
DT Genetic Miner	0,465	0,830	na	na	0,022	0,693	0,898	0,297	na	16h:50m:18s
Flower	1,000	1,000	0,659	0,000	1,000	0,540	0,000	0,049	0,110	1s
Genetic Miner	0,288	0,873	0,935	0,540	0,001	0,731	0,882	0,276	na	6u:15m:14s
HeuristicsMiner	0,794	0,703	0,226	0,000	0,000	1,000	0,898	0,262	na	1s
ILP Miner	0,998	1,000	na	na	1,000	1,000	0,709	0,150	0,315	1m:29s

Table A.4: Accuracy metrics for SIM data set

	Recall					Precision				Run time
	f	r_B^p	$PF_{Complete}$	PM	set	$a_B^{'}$	s_B^n	p_B	etc_P	
AGNEsMiner	0,689	0,762	na	na	0,087	0,614	0,892	0,339	na	36m:33s
α^+	0,898	0,835	na	na	0,000	0,640	0,421	0,095	1	1s
α^{++}	0,750	0,523	na	na	0,000	0,609	0,732	0,125	1	1s
DT Genetic Miner	0,752	0,974	na	na	0,014	0,681	0,900	0,415	na	22h:8m:26s
Flower	1,000	1,000	0,332	0,000	1,000	0,360	0,000	0,068	0,204	1s
Genetic Miner	0,473	0,810	0,902	0,054	0,065	0,707	0,821	0,249	na	1d:8h:28m:55s
HeuristicsMiner	0,970	0,675	0,000	0,000	0,907	0,789	0,950	0,498	na	1s
ILP Miner	0,980	0,792	na	na	1,000	0,834	0,856	0,286	0,605	53s

Table A.5: Accuracy metrics for XBM data set

	Recall					Precision				Run time
	f	r_B^p	$PF_{Complete}$	PM	set	$a_B^{'}$	s_B^n	p_B	etc_P	
AGNEsMiner	0,765	0,862	na	na	0,666	0,363	0,964	0,344	na	3h:24m:17s
α^+	0,927	0,702	na	na	0,065	0,598	0,971	0,346	0,424	2s
α^{++}	0,750	0,614	0,526	0,034	0,000	0,598	0,992	0,630	1,000	1s
DT Genetic Miner	0,795	0,894	0,846	0,000	0,506	0,431	0,950	0,280	na	5h:33m:44s
Flower	1,000	1,000	0,355	0,000	1,000	0,100	0,000	0,022	0,060	1s
Genetic Miner	0,684	0,940	0,829	0,026	0,467	0,707	0,970	0,407	na	5h:9m:45s
HeuristicsMiner	0,985	0,840	0,784	0,137	0,836	0,856	0,995	0,776	na	1s
ILP Miner	0,958	0,953	na	na	1,000	0,591	0,971	0,420	0,694	1,000

Table A.6: Accuracy metrics for XOA data set

	Recall					Precision				Run time
	f	r_B^p	$PF_{Complete}$	PM	set	$a_B^{'}$	s_B^n	p_B	etc_P	
AGNEsMiner	0,724	0,735	0,483	0,000	0,170	0,342	0,915	0,051	1,000	6h:11m:44s
α^+	na	na	na	na	na	na	na	na	na	na
α^{++}	na	na	na	na	na	na	na	na	na	na
DT Genetic Miner	0,686	0,681	na	na	0,074	0,837	0,928	0,056	na	2d:3h:43m:34s
Flower	1,000	1,000	0,352	0,000	1,000	0,000	0,074	0,007	0,063	1s
Genetic Miner	0,441	0,618	0,590	0,000	0,044	0,764	0,937	0,058	na	2d:8h:11m:45s
HeuristicsMiner	0,779	0,779	0,728	0,089	0,172	0,429	0,989	0,305	na	21s
ILP Miner	0,977	0,940	na	na	1,000	0,115	0,978	0,215	0,751	6m:7s

Table A.7: Accuracy metrics for XNB data set

	Recall					Precision				Run time
	f	r_B^p	$PF_{Complete}$	PM	set	$a_B^{'}$	s_B^n	p_B	etc_P	
AGNEsMiner	0,967	0,953	0,953	0,785	0,785	0,635	0,974	0,763	0,734	4m:21s
α^+	0,710	0,638	na	na	0,117	0,737	0,920	0,412	0,706	1s
α^{++}	0,745	0,708	0,576	0,020	0,117	0,645	0,913	0,417	1,000	1s
DT Genetic Miner	0,936	0,983	0,971	0,085	0,899	0,654	0,949	0,628	na	1h:11m:58s
Flower	1,000	1,000	0,324	0,000	1,000	0,180	0,000	0,081	0,162	1s
Genetic Miner	0,995	0,999	0,960	0,741	0,995	0,648	0,966	0,721	na	1u:2m:22s
HeuristicsMiner	0,986	0,853	0,834	0,114	0,811	0,694	0,983	0,817	na	1s
ILP Miner	0,976	0,934	na	na	1,000	0,400	0,924	0,520	0,706	16s

Table A.8: Accuracy metrics for XAO data set

Appendix B

Detailed Comprehensibility Results for the Real-Life Event Logs

This appendix presents the individual comprehensibility results for each of the real-life event logs gauging the performance of the eight process discovery techniques as assessed in Chapter 3.

	transitions	places	arcs	AND-Joins	AND-Splits	OR-Joins	OR-Splits	ECaM	ECyM	SM
AGNEsMiner	79	131	2508	31	27	90	94	605	∞	16590
α^{+}	42	4	10	0	0	1	1	3	153	87
α^{++}	42	20	180	28	28	17	17	58	62	528
DT Genetic Miner	212	120	476	25	16	18	19	298	∞	14555
Flower	44	3	88	0	0	1	1	3	44	44
Genetic Miner	363	1633	5865	251	260	755	498	2701	∞	6555780
HeuristicsMiner	163	94	535	49	58	40	52	271	153171	148330
ILP Miner	42	329	6224	40	41	328	327	3111	∞	588

Table B.1: Comprehensibility metrics for UFM data set

	transitions	places	arcs	AND-Joins	AND-Splits	OR-Joins	OR-Splits	ECaM	ECyM	SM
AGNEsMiner	23	17	49	1	1	5	7	22	∞	464
α^{+}	23	19	74	11	7	9	9	31	∞	644
α^{++}	na	na	na	na	na	na	na	na	na	na
DT Genetic Miner	32	35	91	8	7	10	5	44	328	2688
Flower	25	3	50	0	0	1	1	3	25	25
Genetic Miner	47	35	110	11	5	10	11	56	165	6909
HeuristicsMiner	32	28	80	8	5	9	7	38	42	2605
ILP Miner	23	16	177	13	15	14	12	83	∞	966

Table B.2: Comprehensibility metrics for P2P data set

	transitions	places	arcs	AND-Joins	AND-Splits	OR-Joins	OR-Splits	ECaM	ECyM	SM
AGNEsMiner	36	27	123	23	13	9	12	63	7848	11088
α^{+}	18	50	192	15	15	19	31	100	2	1638
α^{++}	18	66	318	15	15	37	57	176	3	2646
DT Genetic Miner	62	48	168	11	12	11	17	81	∞	6076
Flower	20	3	40	0	0	1	1	3	20	20
Genetic Miner	72	71	258	17	12	27	19	122	∞	26462
HeuristicsMiner	55	33	153	26	17	16	15	81	∞	11341
ILP Miner	18	26	406	16	12	25	25	109	∞	8705

Table B.3: Comprehensibility metrics for KHD data set

	transitions	places	arcs	AND-Joins	AND-Splits	OR-Joins	OR-Splits	ECaM	ECyM	SM
AGNEsMiner	60	42	158	6	0	8	5	26	2455	2646
α^{+}	22	55	187	15	17	30	17	76	∞	10780
α^{++}	22	115	604	18	19	103	82	221	∞	13552
DT Genetic Miner	131	227	605	76	68	52	44	293	184713	74277
Flower	24	3	48	0	0	1	1	3	24	24
Genetic Miner	81	57	211	8	12	23	15	106	0	22401
HeuristicsMiner	42	29	102	6	8	11	11	46	∞	842
ILP Miner	22	24	341	14	20	22	22	122	∞	924

Table B.4: Comprehensibility metrics for SIM data set

	transitions	places	arcs	AND-Joins	AND-Splits	OR-Joins	OR-Splits	ECaM	ECyM	SM
AGNEsMiner	24	18	130	20	13	11	9	51	173	4032
α^{+}	18	11	29	4	3	3	4	16	∞	756
α^{++}	18	19	80	8	9	13	14	36	29	900
DT Genetic Miner	32	24	79	5	4	6	7	38	∞	896
Flower	20	3	40	0	0	1	1	3	0	20
Genetic Miner	54	59	196	24	17	16	22	99	∞	36358
HeuristicsMiner	40	33	143	21	15	12	13	62	983	2583
ILP Miner	18	27	373	16	17	25	25	181	∞	126

Table B.5: Comprehensibility metrics for XBM data set

	transitions	places	arcs	AND-Joins	AND-Splits	OR-Joins	OR-Splits	ECaM	ECyM	SM
AGNEsMiner	64	41	166	18	14	12	20	71	12004	25375
α^{+}	49	96	513	25	38	57	26	137	82	1088476
α^{++}	49	152	996	32	41	115	91	453	40	147001
DT Genetic Miner	51	40	128	5	10	10	5	54	9940	10290
Flower	51	3	102	0	0	1	1	3	51	51
Genetic Miner	60	64	201	22	10	10	23	95	∞	67432
HeuristicsMiner	75	60	236	24	36	21	24	95	508	48777
ILP Miner	49	46	362	25	32	36	29	158	∞	18817

Table B.6: Comprehensibility metrics for XOA data set

	transitions	places	arcs	AND-Joins	AND-Splits	OR-Joins	OR-Splits	ECaM	ECyM	SM
AGNEsMiner	178	115	392	4	32	18	22	na	na	na
α^{+}	na	na	na	na	na	na	na	na	na	na
α^{++}	na	na	na	na	na	na	na	na	na	na
DT Genetic Miner	224	149	529	33	27	16	25	na	na	na
Flower	165	3	336	0	0	1	1	3	165	165
Genetic Miner	269	347	1074	117	109	78	94	524	698	288099
HeuristicsMiner	287	189	1999	79	87	72	72	na	na	na
ILP Miner	163	175	5096	114	123	139	137	3103	∞	30996

Table B.7: Comprehensibility metrics for XNB data set

	transitions	places	arcs	AND-Joins	AND-Splits	OR-Joins	OR-Splits	ECaM	ECyM	SM
AGNEsMiner	16	13	48	5	6	8	4	21	41	1051
α^{+}	14	19	65	8	9	10	6	26	25	911
α^{++}	14	18	69	8	10	12	6	25	34	1912
DT Genetic Miner	30	19	63	1	2	6	6	31	166	120
Flower	16	3	32	0	0	1	1	3	16	16
Genetic Miner	30	21	73	6	6	7	9	36	∞	610
HeuristicsMiner	23	19	66	7	8	6	8	32	46	966
ILP Miner	14	11	84	5	10	7	8	28	∞	911

Table B.8: Comprehensibility metrics for XAO data set

Appendix C

Detailed Results of the Experimental Evaluation of ActiTraC with the Artificial Event Logs

The following three tables present the detailed results of the experimental evaluation of the ActiTraC algorithms based on artificially generated event logs as described in Chapter 4. Note that for the different trace clustering techniques, different parameter settings are taken into account. As to the notation, the following symbols are used:

- H_A denotes the Average Entropy,
- $PM_{W.A.}$ denotes the Weighted Average Parsing Measure,
- PM_A denotes the Average Parsing Measure,
- $R_{W.A.}$ denotes the Weighted Average Recall,
- $P_{W.A.}$ denotes the Weighted Average Precision,
- $F1_{W.A.}$ denotes the Weighted Average F1-score,
- $F2_{W.A.}$ denotes the Weighted Average F2-score,
- P/T-CD$_A$ denotes the Average Place/Transition Connection Degree, and
- P/T-CD$_{W.A.}$ denotes the Weighted Average Place/Transition Connection Degree.

Event log with exponentially distributed frequencies of dpi's

Technique	Parameter settings	H_A	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	1,512	0,621	0,152	0,945	0,984	0,964	0,952	3,140	3,159	00:13:29,120
ActiTraC^{freq}	mcs50tf100	1,512	0,621	0,152	0,945	0,984	0,964	0,952	3,140	3,159	00:13:17,893
ActiTraC^{freq}	mcs100tf100	1,512	0,621	0,152	0,945	0,984	0,964	0,952	3,140	3,159	00:13:17,843
ActiTraC^{freq}	mcs25tf95	1,478	0,165	0,032	0,937	0,923	0,929	0,934	3,252	3,246	00:01:20,273
ActiTraC^{freq}	mcs50tf95	1,474	0,007	0,010	0,938	0,914	0,926	0,933	3,322	3,371	00:14:46,769
ActiTraC^{freq}	mcs100tf95	1,474	0,007	0,010	0,938	0,914	0,926	0,933	3,322	3,371	00:14:47,057
ActiTraC^{MRA}	w25mcs25tf100	1,481	0,412	0,118	0,917	0,961	0,935	0,923	2,789	3,124	00:06:33,933
ActiTraC^{MRA}	w25mcs50tf100	1,481	0,412	0,118	0,917	0,961	0,935	0,923	2,789	3,124	00:06:34,604
ActiTraC^{MRA}	w25mcs100tf100	1,481	0,412	0,118	0,917	0,961	0,935	0,923	2,789	3,124	00:06:34,822
ActiTraC^{MRA}	w50mcs25tf100	1,529	0,341	0,058	0,839	0,695	0,759	0,804	2,733	2,770	00:05:53,343
ActiTraC^{MRA}	w50mcs50tf100	1,529	0,341	0,058	0,839	0,695	0,759	0,804	2,733	2,770	00:05:53,123
ActiTraC^{MRA}	w50mcs100tf100	1,529	0,341	0,058	0,839	0,695	0,759	0,804	2,733	2,770	00:05:52,983
ActiTraC^{MRA}	w100mcs25tf100	1,529	0,341	0,058	0,839	0,695	0,759	0,804	2,733	2,770	00:05:37,234
ActiTraC^{MRA}	w100mcs50tf100	1,529	0,341	0,058	0,839	0,695	0,759	0,804	2,733	2,770	00:05:37,172
ActiTraC^{MRA}	w100mcs100tf100	1,529	0,341	0,058	0,839	0,695	0,759	0,804	2,733	2,770	00:05:37,249
ActiTraC^{MRA}	w25mcs25tf95	1,506	0,068	0,020	0,902	0,727	0,798	0,855	2,961	2,965	00:04:31,958
ActiTraC^{MRA}	w25mcs50tf95	1,537	0,016	0,018	0,896	0,719	0,791	0,848	2,975	2,967	00:06:54,829
ActiTraC^{MRA}	w25mcs100tf95	1,537	0,016	0,018	0,896	0,719	0,791	0,848	2,975	2,967	00:06:54,938
ActiTraC^{MRA}	w50mcs25tf95	1,521	0,100	0,005	0,884	0,758	0,816	0,855	2,780	2,815	00:06:05,583
ActiTraC^{MRA}	w50mcs50tf95	1,521	0,100	0,005	0,884	0,758	0,816	0,855	2,780	2,815	00:06:05,755
ActiTraC^{MRA}	w50mcs100tf95	1,521	0,100	0,005	0,884	0,758	0,816	0,855	2,780	2,815	00:06:05,770
ActiTraC^{MRA}	w100mcs25tf95	1,498	0,125	0,008	0,893	0,903	0,894	0,892	2,790	2,929	00:05:44,688
ActiTraC^{MRA}	w100mcs50tf95	1,469	0,077	0,010	0,881	0,900	0,887	0,882	2,743	2,833	00:06:05,007
ActiTraC^{MRA}	w100mcs100tf95	1,469	0,077	0,010	0,881	0,900	0,887	0,882	2,743	2,833	00:06:04,866
MR	Nominal-EuclideanDistance	1,556	0,000	0,000	0,836	0,774	0,803	0,823	2,963	2,921	00:00:29,050
MR	Nominal-FscoreSimilarity	1,518	0,000	0,000	0,823	0,783	0,799	0,813	3,028	3,038	00:00:17,774
MR	Numeric-EuclideanDistance	1,556	0,000	0,000	0,836	0,774	0,803	0,823	2,963	2,921	00:00:30,205
MR	Numeric-FscoreSimilarity	1,518	0,000	0,000	0,823	0,783	0,799	0,813	3,028	3,038	00:00:20,025
MRA	Nominal-EuclideanDistance	1,524	0,000	0,000	0,853	0,733	0,787	0,825	2,924	2,960	00:00:19,511
MRA	Nominal-FscoreSimilarity	1,530	0,005	0,002	0,848	0,778	0,810	0,832	2,967	3,044	00:00:15,830
MRA	Numeric-EuclideanDistance	1,524	0,000	0,000	0,853	0,733	0,787	0,825	2,924	2,960	00:00:20,028
MRA	Numeric-FscoreSimilarity	1,530	0,005	0,002	0,848	0,778	0,810	0,832	2,967	3,044	00:00:15,607
GED	-	1,498	0,000	0,000	0,885	0,772	0,821	0,858	3,082	3,114	00:00:23,949
LED	-	1,442	0,025	0,003	0,859	0,732	0,790	0,830	2,898	2,972	00:00:14,732
BOA	Nominal-EuclideanDistance	1,533	0,000	0,000	0,830	0,679	0,745	0,793	3,029	3,020	00:00:12,420
BOA	Nominal-FscoreSimilarity	1,481	0,000	0,002	0,887	0,795	0,838	0,867	3,051	3,080	00:00:13,216
BOA	Numeric-EuclideanDistance	1,533	0,000	0,000	0,830	0,679	0,745	0,793	3,029	3,020	00:00:12,826
BOA	Numeric-FscoreSimilarity	1,481	0,000	0,002	0,887	0,795	0,838	0,867	3,051	3,080	00:00:13,263
3-gram	Nominal-EuclideanDistance	1,531	0,000	0,000	0,800	0,717	0,755	0,781	2,992	2,982	00:00:17,137
3-gram	Nominal-FscoreSimilarity	1,430	0,000	0,000	0,795	0,812	0,789	0,790	3,051	3,033	00:00:13,169
3-gram	Numeric-EuclideanDistance	1,531	0,000	0,000	0,800	0,717	0,755	0,781	2,992	2,982	00:00:17,199
3-gram	Numeric-FscoreSimilarity	1,430	0,000	0,000	0,795	0,812	0,789	0,790	3,051	3,033	00:00:12,857
Markov	-	1,456	0,000	0,000	0,822	0,747	0,780	0,804	3,153	3,170	00:29:05,000
Random	-	1,532	0,000	0,001	0,828	0,747	0,782	0,808	3,047	3,045	00:00:07,055
No clustering	-	NA	0,000	0,000	0,826	0,718	0,768	0,802	2,949	2,949	NA

Table C.1: Results of different trace clustering techniques with varying parameter settings for the artificial event log with exponentially distributed frequencies in terms of dpi's.

Event log with linearly distributed frequencies of dpi's

Technique	Parameter settings	H_A	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	1,569	0,309	0,255	0,910	0,990	0,948	0,925	3,510	3,660	00:03:32,311
ActiTraC^{freq}	mcs50tf100	1,569	0,309	0,255	0,910	0,990	0,948	0,925	3,510	3,660	00:03:32,514
ActiTraC^{freq}	mcs100tf100	1,569	0,309	0,255	0,910	0,990	0,948	0,925	3,510	3,660	00:03:39,827
ActiTraC^{freq}	mcs25tf95	1,572	0,036	0,032	0,959	0,949	0,954	0,957	3,637	3,683	00:03:06,222
ActiTraC^{freq}	mcs50tf95	1,575	0,034	0,022	0,950	0,962	0,956	0,952	3,454	3,385	00:06:02,995
ActiTraC^{freq}	mcs100tf95	1,577	0,034	0,022	0,950	0,962	0,956	0,952	3,454	3,385	00:06:54,049
ActiTraC^{MRA}	w25mcs25tf100	1,560	0,275	0,240	0,903	0,966	0,933	0,915	3,366	3,612	00:02:15,090
ActiTraC^{MRA}	w25mcs50tf100	1,560	0,275	0,240	0,903	0,966	0,933	0,915	3,366	3,612	00:02:15,074
ActiTraC^{MRA}	w25mcs100tf100	1,560	0,275	0,240	0,903	0,966	0,933	0,915	3,366	3,612	00:02:14,934
ActiTraC^{MRA}	w50mcs25tf100	1,567	0,219	0,182	0,885	0,991	0,934	0,904	3,281	3,279	00:01:57,780
ActiTraC^{MRA}	w50mcs50tf100	1,567	0,219	0,182	0,885	0,991	0,934	0,904	3,281	3,279	00:01:57,516
ActiTraC^{MRA}	w50mcs100tf100	1,567	0,219	0,182	0,885	0,991	0,934	0,904	3,281	3,279	00:01:57,843
ActiTraC^{MRA}	w100mcs25tf100	1,561	0,228	0,188	0,886	0,990	0,935	0,905	3,222	3,400	00:01:48,985
ActiTraC^{MRA}	w100mcs50tf100	1,561	0,228	0,188	0,886	0,990	0,935	0,905	3,222	3,400	00:01:48,736
ActiTraC^{MRA}	w100mcs100tf100	1,561	0,228	0,188	0,886	0,990	0,935	0,905	3,222	3,400	00:01:48,923
ActiTraC^{MRA}	w25mcs25tf95	1,567	0,032	0,025	0,932	0,973	0,952	0,940	3,397	3,424	00:03:55,467
ActiTraC^{MRA}	w25mcs50tf95	1,567	0,032	0,025	0,932	0,973	0,952	0,940	3,397	3,424	00:03:55,453
ActiTraC^{MRA}	w25mcs100tf95	1,567	0,032	0,025	0,932	0,973	0,952	0,940	3,397	3,424	00:03:55,421
ActiTraC^{MRA}	w50mcs25tf95	1,581	0,017	0,012	0,945	0,928	0,936	0,941	3,569	3,595	00:04:25,642
ActiTraC^{MRA}	w50mcs50tf95	1,581	0,017	0,012	0,945	0,928	0,936	0,941	3,569	3,595	00:04:25,424
ActiTraC^{MRA}	w50mcs100tf95	1,581	0,017	0,012	0,945	0,928	0,936	0,941	3,569	3,595	00:04:25,798
ActiTraC^{MRA}	w100mcs25tf95	1,571	0,035	0,028	0,926	0,895	0,910	0,920	3,539	3,635	00:03:08,639
ActiTraC^{MRA}	w100mcs50tf95	1,571	0,035	0,028	0,926	0,895	0,910	0,920	3,539	3,635	00:03:08,639
ActiTraC^{MRA}	w100mcs100tf95	1,571	0,035	0,028	0,926	0,895	0,910	0,920	3,539	3,635	00:03:08,640
MR	Nominal-EuclideanDistance	1,582	0,000	0,000	0,865	0,775	0,817	0,845	2,918	2,916	00:00:29,100
MR	Nominal-FscoreSimilarity	1,572	0,001	0,002	0,859	0,697	0,769	0,821	2,975	2,966	00:00:16,432
MR	Numeric-EuclideanDistance	1,582	0,000	0,000	0,865	0,775	0,817	0,845	2,918	2,916	00:00:27,741
MR	Numeric-FscoreSimilarity	1,572	0,001	0,002	0,859	0,697	0,769	0,821	2,975	2,966	00:00:16,089
MRA	Nominal-EuclideanDistance	1,582	0,000	0,000	0,826	0,733	0,776	0,805	3,009	2,918	00:00:20,028
MRA	Nominal-FscoreSimilarity	1,577	0,017	0,013	0,853	0,818	0,834	0,845	3,023	2,980	00:00:15,809
MRA	Numeric-EuclideanDistance	1,582	0,000	0,000	0,826	0,733	0,776	0,805	3,009	2,918	00:00:20,324
MRA	Numeric-FscoreSimilarity	1,577	0,017	0,013	0,853	0,818	0,834	0,845	3,023	2,980	00:00:15,856
GED	-	1,577	0,018	0,015	0,896	0,791	0,839	0,872	2,907	2,896	00:00:23,339
LED	-	1,578	0,009	0,010	0,852	0,808	0,828	0,842	2,993	2,979	00:00:13,263
BOA	Nominal-EuclideanDistance	1,583	0,002	0,002	0,864	0,790	0,824	0,847	2,986	2,972	00:00:13,170
BOA	Nominal-FscoreSimilarity	1,577	0,006	0,010	0,907	0,793	0,846	0,881	2,907	2,918	00:00:13,139
BOA	Numeric-EuclideanDistance	1,583	0,002	0,002	0,864	0,790	0,824	0,847	2,986	2,972	00:00:13,434
BOA	Numeric-FscoreSimilarity	1,577	0,006	0,010	0,907	0,793	0,846	0,881	2,907	2,918	00:00:12,950
3-gram	Nominal-EuclideanDistance	1,576	0,006	0,007	0,880	0,741	0,803	0,847	3,038	2,948	00:00:17,559
3-gram	Nominal-FscoreSimilarity	1,579	0,003	0,005	0,856	0,772	0,812	0,838	2,982	2,962	00:00:14,982
3-gram	Numeric-EuclideanDistance	1,576	0,006	0,007	0,880	0,741	0,803	0,847	3,038	2,948	00:00:17,308
3-gram	Numeric-FscoreSimilarity	1,579	0,003	0,005	0,856	0,772	0,812	0,838	2,982	2,962	00:00:12,467
Markov	-	1,507	0,001	0,002	0,835	0,777	0,804	0,822	3,203	3,213	01:04:46,000
Random	-	1,580	0,000	0,000	0,843	0,753	0,795	0,823	3,004	3,001	00:00:08,929
No clustering	-	NA	0,009	0,012	0,885	0,811	0,846	0,869	2,941	2,941	NA

Table C.2: Results of different trace clustering techniques with varying parameter settings for the artificial event log with linearly distributed frequencies in terms of dpi's.

Event log with uniformly distributed frequencies of dpi's

Technique	Parameter settings	H_A	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	1,564	0,088	0,088	0,842	0,888	0,862	0,849	3,058	3,099	00:00:56,465
ActiTraC^{freq}	mcs50tf100	1,564	0,088	0,088	0,842	0,888	0,862	0,849	3,058	3,099	00:00:56,435
ActiTraC^{freq}	mcs100tf100	1,564	0,088	0,088	0,842	0,888	0,862	0,849	3,058	3,099	00:00:56,622
ActiTraC^{freq}	mcs25tf95	1,576	0,025	0,025	0,924	0,921	0,922	0,923	3,468	3,623	00:02:44,111
ActiTraC^{freq}	mcs50tf95	1,569	0,022	0,022	0,926	0,917	0,920	0,924	3,440	3,590	00:03:31,563
ActiTraC^{freq}	mcs100tf95	1,569	0,022	0,022	0,926	0,917	0,920	0,924	3,440	3,590	00:03:31,453
ActiTraC^{MRA}	w25mcs25tf100	1,558	0,127	0,127	0,855	0,884	0,867	0,859	3,129	3,116	00:00:57,401
ActiTraC^{MRA}	w25mcs50tf100	1,558	0,127	0,127	0,855	0,884	0,867	0,859	3,129	3,116	00:00:57,292
ActiTraC^{MRA}	w25mcs100tf100	1,558	0,127	0,127	0,855	0,884	0,867	0,859	3,129	3,116	00:00:57,152
ActiTraC^{MRA}	w50mcs25tf100	1,558	0,127	0,127	0,855	0,884	0,867	0,859	3,129	3,116	00:00:57,089
ActiTraC^{MRA}	w50mcs50tf100	1,558	0,127	0,127	0,855	0,884	0,867	0,859	3,129	3,116	00:00:57,121
ActiTraC^{MRA}	w50mcs100tf100	1,558	0,127	0,127	0,855	0,884	0,867	0,859	3,129	3,116	00:00:57,276
ActiTraC^{MRA}	w100mcs25tf100	1,558	0,127	0,127	0,855	0,884	0,867	0,859	3,129	3,116	00:00:58,135
ActiTraC^{MRA}	w100mcs50tf100	1,558	0,127	0,127	0,855	0,884	0,867	0,859	3,129	3,116	00:00:57,947
ActiTraC^{MRA}	w100mcs100tf100	1,558	0,127	0,127	0,855	0,884	0,867	0,859	3,129	3,116	00:00:57,385
ActiTraC^{MRA}	w25mcs25tf95	1,568	0,042	0,042	0,933	0,902	0,917	0,927	3,692	3,695	00:03:00,313
ActiTraC^{MRA}	w25mcs50tf95	1,568	0,042	0,042	0,933	0,902	0,917	0,927	3,692	3,695	00:03:00,452
ActiTraC^{MRA}	w25mcs100tf95	1,568	0,042	0,042	0,933	0,902	0,917	0,927	3,692	3,695	00:03:00,141
ActiTraC^{MRA}	w50mcs25tf95	1,568	0,042	0,042	0,933	0,902	0,917	0,927	3,692	3,695	00:03:00,281
ActiTraC^{MRA}	w50mcs50tf95	1,568	0,042	0,042	0,933	0,902	0,917	0,927	3,692	3,695	00:03:00,234
ActiTraC^{MRA}	w50mcs100tf95	1,568	0,042	0,042	0,933	0,902	0,917	0,927	3,692	3,695	00:03:00,842
ActiTraC^{MRA}	w100mcs25tf95	1,568	0,042	0,042	0,933	0,902	0,917	0,927	3,692	3,695	00:03:03,369
ActiTraC^{MRA}	w100mcs50tf95	1,568	0,042	0,042	0,933	0,902	0,917	0,927	3,692	3,695	00:03:01,638
ActiTraC^{MRA}	w100mcs100tf95	1,568	0,042	0,042	0,933	0,902	0,917	0,927	3,692	3,695	00:03:00,562
MR	Nominal-EuclideanDistance	1,584	0,000	0,000	0,863	0,806	0,833	0,851	2,959	2,947	00:00:27,381
MR	Nominal-FscoreSimilarity	1,575	0,005	0,005	0,856	0,768	0,809	0,836	2,941	2,926	00:00:20,712
MR	Numeric-EuclideanDistance	1,584	0,000	0,000	0,863	0,806	0,833	0,851	2,959	2,947	00:00:29,803
MR	Numeric-FscoreSimilarity	1,575	0,005	0,005	0,856	0,768	0,809	0,836	2,941	2,926	00:00:16,932
MRA	Nominal-EuclideanDistance	1,581	0,000	0,000	0,831	0,758	0,792	0,815	3,029	2,903	00:00:19,528
MRA	Nominal-FscoreSimilarity	1,577	0,010	0,010	0,865	0,814	0,838	0,853	3,060	2,970	00:00:14,559
MRA	Numeric-EuclideanDistance	1,581	0,000	0,000	0,831	0,758	0,792	0,815	3,029	2,903	00:00:19,262
MRA	Numeric-FscoreSimilarity	1,577	0,010	0,010	0,865	0,814	0,838	0,853	3,060	2,970	00:00:15,326
GED	-	1,581	0,017	0,017	0,886	0,815	0,848	0,870	2,957	2,963	00:00:23,574
LED	-	1,576	0,010	0,010	0,865	0,798	0,829	0,850	3,006	2,966	00:00:13,451
BOA	Nominal-EuclideanDistance	1,580	0,003	0,003	0,866	0,771	0,814	0,844	2,992	2,981	00:00:12,529
BOA	Nominal-FscoreSimilarity	1,578	0,010	0,010	0,901	0,786	0,839	0,875	2,896	2,916	00:00:13,060
BOA	Numeric-EuclideanDistance	1,580	0,003	0,003	0,866	0,771	0,814	0,844	2,992	2,981	00:00:13,060
BOA	Numeric-FscoreSimilarity	1,578	0,010	0,010	0,901	0,786	0,839	0,875	2,896	2,916	00:00:12,982
3-gram	Nominal-EuclideanDistance	1,579	0,007	0,007	0,877	0,756	0,812	0,850	2,995	2,937	00:00:17,371
3-gram	Nominal-FscoreSimilarity	1,579	0,008	0,008	0,853	0,720	0,779	0,821	2,973	2,968	00:00:13,248
3-gram	Numeric-EuclideanDistance	1,579	0,007	0,007	0,877	0,756	0,812	0,850	2,995	2,937	00:00:17,169
3-gram	Numeric-FscoreSimilarity	1,579	0,008	0,008	0,853	0,720	0,779	0,821	2,973	2,968	00:00:13,310
Markov	-	1,551	0,002	0,002	0,795	0,825	0,805	0,798	3,286	3,289	00:52:58,000
Random	-	1,581	0,002	0,002	0,862	0,769	0,812	0,841	2,985	2,985	00:00:10,229
No clustering	-	NA	0,023	0,023	0,899	0,807	0,851	0,879	2,946	2,946	NA

Table C.3: Results of different trace clustering techniques with varying parameter settings for the artificial event log with uniformly distributed frequencies in terms of dpi's.

Appendix D

Detailed Results of the Real-Life Experiments of the Comparative Evaluation of the ActiTraC Algorithms

The following twelve tables present the detailed results of the experimental evaluation of the ActiTraC algorithms based on four real-life event logs as described in Chapter 4. Note that for the different trace clustering techniques, different parameter settings are taken into account. Furthermore, the number of clusters is varied from 3 to 5. As to the notation, the following symbols are used:

- $PM_{W.A.}$ denotes the Weighted Average Parsing Measure,
- PM_A denotes the Average Parsing Measure,
- $R_{W.A.}$ denotes the Weighted Average Recall,
- $P_{W.A.}$ denotes the Weighted Average Precision,
- $F1_{W.A.}$ denotes the Weighted Average F1-score,
- $F2_{W.A.}$ denotes the Weighted Average F2-score,
- P/T-CD$_A$ denotes the Average Place/Transition Connection Degree, and
- P/T-CD$_{W.A.}$ denotes the Weighted Average Place/Transition Connection Degree.

Event log KIM - 3 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,927	0,246	0,986	0,978	0,982	0,984	2,774	2,733	00:04:42,812
ActiTraC^{freq}	mcs50tf100	0,938	0,313	0,989	0,971	0,979	0,985	3,022	2,864	00:03:45,775
ActiTraC^{freq}	mcs100tf100	0,952	0,371	0,990	0,977	0,983	0,987	2,997	3,298	01:15:13,238
ActiTraC^{freq}	mcs25tf95	0,903	0,209	0,986	0,973	0,979	0,983	2,838	2,755	00:01:08,642
ActiTraC^{freq}	mcs50tf95	0,818	0,090	0,984	0,938	0,959	0,973	3,098	2,885	00:06:36,917
ActiTraC^{freq}	mcs100tf95	0,207	0,016	0,959	0,963	0,958	0,958	2,897	3,349	01:16:33,304
ActiTraC^{MRA}	w25mcs25tf100	0,898	0,175	0,976	0,951	0,963	0,971	2,782	2,821	00:00:41,838
ActiTraC^{MRA}	w25mcs50tf100	0,915	0,221	0,981	0,964	0,972	0,978	2,914	2,912	00:05:58,339
ActiTraC^{MRA}	w25mcs100tf100	0,941	0,358	0,987	0,962	0,974	0,982	2,988	3,135	01:14:21,066
ActiTraC^{MRA}	w50mcs25tf100	0,898	0,175	0,976	0,951	0,963	0,971	2,782	2,821	00:00:42,091
ActiTraC^{MRA}	w50mcs50tf100	0,915	0,221	0,981	0,964	0,972	0,978	2,914	2,912	00:05:56,972
ActiTraC^{MRA}	w50mcs100tf100	0,941	0,358	0,987	0,962	0,974	0,982	2,988	3,135	01:13:30,044
ActiTraC^{MRA}	w100mcs25tf100	0,898	0,175	0,976	0,951	0,963	0,971	2,782	2,821	00:00:42,716
ActiTraC^{MRA}	w100mcs50tf100	0,915	0,221	0,981	0,964	0,972	0,978	2,914	2,912	00:05:59,967
ActiTraC^{MRA}	w100mcs100tf100	0,941	0,358	0,987	0,962	0,974	0,982	2,988	3,135	01:13:00,157
ActiTraC^{MRA}	w25mcs25tf95	0,782	0,101	0,973	0,975	0,974	0,973	2,831	2,952	00:00:48,319
ActiTraC^{MRA}	w25mcs50tf95	0,784	0,099	0,963	0,969	0,965	0,964	3,071	3,100	00:04:58,579
ActiTraC^{MRA}	w25mcs100tf95	0,204	0,010	0,962	0,899	0,926	0,947	3,107	3,302	01:04:23,124
ActiTraC^{MRA}	w50mcs25tf95	0,764	0,060	0,962	0,934	0,946	0,955	2,950	3,009	00:00:54,774
ActiTraC^{MRA}	w50mcs50tf95	0,495	0,033	0,957	0,942	0,946	0,952	3,066	3,190	00:09:23,305
ActiTraC^{MRA}	w50mcs100tf95	0,204	0,010	0,962	0,899	0,926	0,947	3,107	3,302	01:04:01,626
ActiTraC^{MRA}	w100mcs25tf95	0,764	0,060	0,962	0,934	0,946	0,955	2,950	3,009	00:00:55,959
ActiTraC^{MRA}	w100mcs50tf95	0,495	0,033	0,957	0,942	0,946	0,952	3,066	3,190	00:09:23,956
ActiTraC^{MRA}	w100mcs100tf95	0,204	0,010	0,962	0,899	0,926	0,947	3,107	3,302	01:03:20,846
MR	Nominal-EuclideanDistance	0,212	0,005	0,817	0,948	0,871	0,836	3,157	3,364	00:01:40,151
MR	Nominal-FscoreSimilarity	0,195	0,003	0,886	0,895	0,881	0,881	3,216	3,297	00:00:54,439
MR	Numeric-EuclideanDistance	0,212	0,005	0,817	0,948	0,871	0,836	3,157	3,364	00:01:36,634
MR	Numeric-FscoreSimilarity	0,195	0,003	0,886	0,895	0,881	0,881	3,216	3,297	00:00:52,987
MRA	Nominal-EuclideanDistance	0,196	0,004	0,820	0,914	0,859	0,834	3,134	3,277	00:01:12,616
MRA	Nominal-FscoreSimilarity	0,260	0,005	0,878	0,752	0,807	0,847	3,331	3,372	00:00:53,830
MRA	Numeric-EuclideanDistance	0,196	0,004	0,820	0,914	0,859	0,834	3,134	3,277	00:01:15,574
MRA	Numeric-FscoreSimilarity	0,260	0,005	0,878	0,752	0,807	0,847	3,331	3,372	00:00:51,429
GED	-	0,213	0,007	0,829	0,722	0,765	0,799	3,404	3,432	00:01:09,062
LED	-	0,195	0,004	0,878	0,893	0,878	0,875	3,342	3,421	00:00:43,667
BOA	Nominal-EuclideanDistance	0,214	0,006	0,817	0,952	0,874	0,837	3,243	3,445	00:00:36,467
BOA	Nominal-FscoreSimilarity	0,213	0,009	0,836	0,730	0,772	0,806	3,298	3,220	00:00:32,095
BOA	Numeric-EuclideanDistance	0,214	0,006	0,817	0,952	0,874	0,837	3,243	3,445	00:00:41,371
BOA	Numeric-FscoreSimilarity	0,213	0,009	0,836	0,730	0,772	0,806	3,298	3,220	00:00:32,047
3-gram	Nominal-EuclideanDistance	0,215	0,012	0,820	0,981	0,892	0,847	2,865	3,443	00:00:54,631
3-gram	Nominal-FscoreSimilarity	0,215	0,009	0,830	0,756	0,781	0,807	3,253	3,313	00:00:37,030
3-gram	Numeric-EuclideanDistance	0,215	0,012	0,820	0,981	0,892	0,847	2,865	3,443	00:00:57,973
3-gram	Numeric-FscoreSimilarity	0,215	0,009	0,830	0,756	0,781	0,807	3,253	3,313	00:00:36,764
Markov	-	0,277	0,010	0,934	0,961	0,947	0,939	3,173	3,148	07:21:55,000
Random	-	0,214	0,004	0,843	0,776	0,804	0,826	3,272	3,293	00:00:18,038
No clustering	-	0,198	0,006	0,821	0,985	0,895	0,849	3,540	3,540	NA

Table D.1: Results of different trace clustering techniques with varying parameter settings for event log KIM in case of the creation of 3 clusters.

Event log KIM - 4 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,948	0,366	0,991	0,983	0,987	0,989	2,797	2,745	00:04:41,347
ActiTraC^{freq}	mcs50tf100	0,948	0,350	0,991	0,973	0,981	0,987	2,967	2,861	00:04:45,702
ActiTraC^{freq}	mcs100tf100	0,956	0,387	0,991	0,978	0,983	0,988	2,919	3,295	01:13:28,792
ActiTraC^{freq}	mcs25tf95	0,909	0,290	0,989	0,979	0,984	0,987	2,901	2,773	00:01:12,321
ActiTraC^{freq}	mcs50tf95	0,817	0,076	0,987	0,931	0,954	0,972	3,068	2,881	00:06:58,280
ActiTraC^{freq}	mcs100tf95	0,210	0,017	0,959	0,965	0,959	0,958	2,762	3,347	01:14:31,231
ActiTraC^{MRA}	w25mcs25tf100	0,916	0,192	0,981	0,952	0,966	0,974	2,748	2,817	00:02:13,160
ActiTraC^{MRA}	w25mcs50tf100	0,931	0,253	0,986	0,966	0,975	0,981	2,883	2,910	00:07:23,729
ActiTraC^{MRA}	w25mcs100tf100	0,950	0,379	0,990	0,963	0,976	0,984	2,894	3,129	01:13:38,657
ActiTraC^{MRA}	w50mcs25tf100	0,916	0,192	0,981	0,952	0,966	0,974	2,748	2,817	00:02:13,212
ActiTraC^{MRA}	w50mcs50tf100	0,927	0,246	0,985	0,965	0,974	0,980	2,868	2,909	00:06:37,542
ActiTraC^{MRA}	w50mcs100tf100	0,950	0,379	0,990	0,963	0,976	0,984	2,894	3,129	01:13:37,962
ActiTraC^{MRA}	w100mcs25tf100	0,916	0,192	0,981	0,952	0,966	0,974	2,748	2,817	00:02:13,180
ActiTraC^{MRA}	w100mcs50tf100	0,927	0,246	0,985	0,965	0,974	0,980	2,868	2,909	00:06:37,792
ActiTraC^{MRA}	w100mcs100tf100	0,950	0,379	0,990	0,963	0,976	0,984	2,894	3,129	01:13:48,820
ActiTraC^{MRA}	w25mcs25tf95	0,787	0,094	0,978	0,976	0,976	0,977	2,882	2,960	00:01:19,476
ActiTraC^{MRA}	w25mcs50tf95	0,783	0,095	0,971	0,954	0,959	0,965	3,008	3,081	00:06:37,323
ActiTraC^{MRA}	w25mcs100tf95	0,204	0,014	0,962	0,900	0,927	0,947	2,921	3,301	01:03:29,326
ActiTraC^{MRA}	w50mcs25tf95	0,778	0,063	0,976	0,950	0,962	0,970	2,951	3,012	00:01:03,966
ActiTraC^{MRA}	w50mcs50tf95	0,567	0,069	0,969	0,950	0,957	0,963	2,950	3,139	00:09:22,908
ActiTraC^{MRA}	w50mcs100tf95	0,204	0,014	0,962	0,900	0,927	0,947	2,921	3,301	01:03:29,336
ActiTraC^{MRA}	w100mcs25tf95	0,778	0,063	0,976	0,950	0,962	0,970	2,951	3,012	00:01:03,919
ActiTraC^{MRA}	w100mcs50tf95	0,567	0,069	0,969	0,950	0,957	0,963	2,950	3,139	00:09:23,425
ActiTraC^{MRA}	w100mcs100tf95	0,204	0,014	0,962	0,900	0,927	0,947	2,921	3,301	01:04:26,909
MR	Nominal-EuclideanDistance	0,212	0,005	0,818	0,949	0,872	0,836	3,005	3,357	00:01:51,065
MR	Nominal-FscoreSimilarity	0,197	0,004	0,889	0,821	0,847	0,870	3,261	3,319	00:01:12,398
MR	Numeric-EuclideanDistance	0,212	0,005	0,818	0,949	0,872	0,836	3,005	3,357	00:01:38,805
MR	Numeric-FscoreSimilarity	0,197	0,004	0,889	0,821	0,847	0,870	3,261	3,319	00:01:00,123
MRA	Nominal-EuclideanDistance	0,196	0,006	0,821	0,914	0,860	0,835	3,009	3,275	00:01:26,031
MRA	Nominal-FscoreSimilarity	0,260	0,008	0,880	0,758	0,812	0,851	3,172	3,365	00:00:58,062
MRA	Numeric-EuclideanDistance	0,196	0,006	0,821	0,914	0,860	0,835	3,009	3,275	00:01:13,278
MRA	Numeric-FscoreSimilarity	0,260	0,008	0,880	0,758	0,812	0,851	3,172	3,365	00:00:50,008
GED	-	0,212	0,006	0,828	0,684	0,745	0,791	3,324	3,441	00:01:10,467
LED	-	0,195	0,004	0,879	0,895	0,879	0,876	3,193	3,419	00:00:44,042
BOA	Nominal-EuclideanDistance	0,214	0,006	0,816	0,945	0,870	0,834	3,115	3,434	00:00:38,717
BOA	Nominal-FscoreSimilarity	0,244	0,020	0,844	0,854	0,836	0,836	3,229	3,180	00:00:31,798
BOA	Numeric-EuclideanDistance	0,214	0,006	0,816	0,945	0,870	0,834	3,115	3,434	00:00:38,404
BOA	Numeric-FscoreSimilarity	0,244	0,020	0,844	0,854	0,836	0,836	3,229	3,180	00:00:32,297
3-gram	Nominal-EuclideanDistance	0,215	0,012	0,828	0,783	0,792	0,809	3,007	3,434	00:00:50,976
3-gram	Nominal-FscoreSimilarity	0,196	0,006	0,862	0,773	0,806	0,836	3,239	3,256	00:00:33,562
3-gram	Numeric-EuclideanDistance	0,215	0,012	0,828	0,783	0,792	0,809	3,007	3,434	00:00:52,319
3-gram	Numeric-FscoreSimilarity	0,196	0,006	0,862	0,773	0,806	0,836	3,239	3,256	00:00:35,764
Markov	-	0,452	0,014	0,889	0,786	0,833	0,866	3,122	3,087	06:40:26,000
Random	-	0,261	0,006	0,837	0,814	0,823	0,831	3,148	3,150	00:00:17,955
No clustering	-	0,198	0,006	0,821	0,985	0,895	0,849	3,540	3,540	NA

Table D.2: Results of different trace clustering techniques with varying parameter settings for event log KIM in case of the creation of 4 clusters.

Event log KIM - 5 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,950	0,382	0,991	0,984	0,987	0,990	2,780	2,745	00:05:13,924
ActiTraC^{freq}	mcs50tf100	0,952	0,361	0,992	0,973	0,982	0,987	2,879	2,858	00:05:09,413
ActiTraC^{freq}	mcs100tf100	0,960	0,406	0,991	0,977	0,984	0,988	2,893	3,293	01:16:27,375
ActiTraC^{freq}	mcs25tf95	0,910	0,289	0,991	0,973	0,980	0,986	2,920	2,774	00:01:48,906
ActiTraC^{freq}	mcs50tf95	0,818	0,081	0,987	0,932	0,954	0,972	2,988	2,879	00:07:09,887
ActiTraC^{freq}	mcs100tf95	0,211	0,024	0,959	0,965	0,959	0,958	2,698	3,347	01:15:48,312
ActiTraC^{MRA}	w25mcs25tf100	0,934	0,262	0,986	0,964	0,975	0,982	2,780	2,819	00:03:52,788
ActiTraC^{MRA}	w25mcs50tf100	0,936	0,267	0,987	0,965	0,975	0,982	2,848	2,909	00:07:50,685
ActiTraC^{MRA}	w25mcs100tf100	0,954	0,396	0,990	0,963	0,976	0,984	2,880	3,128	01:14:52,742
ActiTraC^{MRA}	w50mcs25tf100	0,934	0,262	0,986	0,964	0,975	0,982	2,780	2,819	00:03:53,805
ActiTraC^{MRA}	w50mcs50tf100	0,942	0,305	0,988	0,967	0,977	0,984	2,904	2,913	00:07:32,236
ActiTraC^{MRA}	w50mcs100tf100	0,954	0,396	0,990	0,963	0,976	0,984	2,880	3,128	01:15:01,277
ActiTraC^{MRA}	w100mcs25tf100	0,934	0,262	0,986	0,964	0,975	0,982	2,780	2,819	00:03:52,420
ActiTraC^{MRA}	w100mcs50tf100	0,942	0,305	0,988	0,967	0,977	0,984	2,904	2,913	00:07:30,875
ActiTraC^{MRA}	w100mcs100tf100	0,954	0,396	0,990	0,963	0,976	0,984	2,880	3,128	01:15:05,213
ActiTraC^{MRA}	w25mcs25tf95	0,789	0,096	0,980	0,970	0,974	0,977	2,874	2,961	00:02:25,197
ActiTraC^{MRA}	w25mcs50tf95	0,784	0,091	0,974	0,954	0,960	0,967	2,982	3,080	00:08:52,808
ActiTraC^{MRA}	w25mcs100tf95	0,204	0,017	0,962	0,900	0,927	0,947	2,830	3,301	01:05:03,532
ActiTraC^{MRA}	w50mcs25tf95	0,795	0,073	0,977	0,947	0,960	0,969	2,926	3,006	00:01:53,887
ActiTraC^{MRA}	w50mcs50tf95	0,569	0,070	0,970	0,950	0,957	0,964	2,920	3,139	00:09:53,106
ActiTraC^{MRA}	w50mcs100tf95	0,204	0,017	0,962	0,900	0,927	0,947	2,830	3,301	01:04:25,885
ActiTraC^{MRA}	w100mcs25tf95	0,795	0,073	0,977	0,947	0,960	0,969	2,926	3,006	00:01:53,527
ActiTraC^{MRA}	w100mcs50tf95	0,569	0,070	0,970	0,950	0,957	0,964	2,920	3,139	00:09:53,037
ActiTraC^{MRA}	w100mcs100tf95	0,204	0,017	0,962	0,900	0,927	0,947	2,830	3,301	01:04:27,338
MR	Nominal-EuclideanDistance	0,212	0,005	0,823	0,949	0,873	0,839	2,929	3,346	00:01:49,456
MR	Nominal-FscoreSimilarity	0,197	0,005	0,891	0,820	0,847	0,870	3,123	3,297	00:00:56,625
MR	Numeric-EuclideanDistance	0,212	0,005	0,823	0,949	0,873	0,839	2,929	3,346	00:01:38,461
MR	Numeric-FscoreSimilarity	0,197	0,005	0,891	0,820	0,847	0,870	3,123	3,297	00:00:53,939
MRA	Nominal-EuclideanDistance	0,196	0,006	0,821	0,910	0,857	0,833	2,961	3,252	00:01:21,299
MRA	Nominal-FscoreSimilarity	0,260	0,008	0,882	0,762	0,816	0,854	3,076	3,327	00:00:56,525
MRA	Numeric-EuclideanDistance	0,196	0,006	0,821	0,910	0,857	0,833	2,961	3,252	00:01:14,075
MRA	Numeric-FscoreSimilarity	0,260	0,008	0,882	0,762	0,816	0,854	3,076	3,327	00:00:51,335
GED	-	0,212	0,006	0,827	0,684	0,744	0,789	3,207	3,318	00:01:09,889
LED	-	0,195	0,003	0,879	0,738	0,800	0,845	3,250	3,719	00:00:45,135
BOA	Nominal-EuclideanDistance	0,213	0,009	0,833	0,699	0,755	0,798	3,198	3,493	00:00:36,546
BOA	Nominal-FscoreSimilarity	0,244	0,021	0,842	0,841	0,829	0,832	3,197	3,196	00:00:31,750
BOA	Numeric-EuclideanDistance	0,213	0,009	0,833	0,699	0,755	0,798	3,198	3,493	00:00:36,779
BOA	Numeric-FscoreSimilarity	0,244	0,021	0,842	0,841	0,829	0,832	3,197	3,196	00:01:18,979
3-gram	Nominal-EuclideanDistance	0,215	0,012	0,908	0,837	0,856	0,882	2,937	3,446	00:00:51,351
3-gram	Nominal-FscoreSimilarity	0,196	0,006	0,862	0,729	0,782	0,826	3,149	3,243	00:00:34,812
3-gram	Numeric-EuclideanDistance	0,215	0,012	0,908	0,837	0,856	0,882	2,937	3,446	00:00:53,631
3-gram	Numeric-FscoreSimilarity	0,196	0,006	0,862	0,729	0,782	0,826	3,149	3,243	00:01:30,770
Markov	-	0,288	0,007	0,936	0,780	0,847	0,897	3,130	3,108	09:50:21,000
Random	-	0,272	0,006	0,850	0,800	0,820	0,836	3,106	3,109	00:00:17,614
No clustering	-	0,198	0,006	0,821	0,985	0,895	0,849	3,540	3,540	NA

Table D.3: Results of different trace clustering techniques with varying parameter settings for event log KIM in case of the creation of 5 clusters.

Event log MCRM - 3 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,743	0,160	0,939	0,632	0,755	0,856	2,528	2,704	00:01:04,693
ActiTraC^{freq}	mcs50tf100	0,743	0,160	0,939	0,632	0,755	0,856	2,528	2,704	00:01:03,013
ActiTraC^{freq}	mcs100tf100	0,743	0,160	0,939	0,632	0,755	0,856	2,528	2,704	00:01:03,620
ActiTraC^{freq}	mcs25tf95	0,613	0,090	0,939	0,619	0,743	0,848	2,603	2,792	00:00:18,503
ActiTraC^{freq}	mcs50tf95	0,140	0,061	0,946	0,593	0,728	0,845	2,714	3,179	00:01:16,949
ActiTraC^{freq}	mcs100tf95	0,140	0,061	0,946	0,593	0,728	0,845	2,714	3,179	00:01:16,152
ActiTraC^{MRA}	w25mcs25tf100	0,699	0,160	0,945	0,729	0,804	0,874	2,235	2,154	00:00:07,846
ActiTraC^{MRA}	w25mcs50tf100	0,720	0,160	0,944	0,724	0,807	0,879	2,454	2,354	00:00:17,045
ActiTraC^{MRA}	w25mcs100tf100	0,725	0,156	0,936	0,609	0,737	0,845	2,559	2,777	00:01:00,382
ActiTraC^{MRA}	w50mcs25tf100	0,699	0,160	0,945	0,729	0,804	0,874	2,235	2,154	00:00:07,862
ActiTraC^{MRA}	w50mcs50tf100	0,720	0,160	0,944	0,724	0,807	0,879	2,454	2,354	00:00:17,016
ActiTraC^{MRA}	w50mcs100tf100	0,725	0,156	0,936	0,609	0,737	0,845	2,559	2,777	00:01:01,430
ActiTraC^{MRA}	w100mcs25tf100	0,699	0,160	0,945	0,729	0,804	0,874	2,235	2,154	00:00:07,935
ActiTraC^{MRA}	w100mcs50tf100	0,720	0,160	0,944	0,724	0,807	0,879	2,454	2,354	00:00:17,128
ActiTraC^{MRA}	w100mcs100tf100	0,725	0,156	0,936	0,609	0,737	0,845	2,559	2,777	00:01:01,653
ActiTraC^{MRA}	w25mcs25tf95	0,691	0,113	0,950	0,700	0,796	0,877	2,425	2,462	00:00:09,953
ActiTraC^{MRA}	w25mcs50tf95	0,726	0,179	0,960	0,661	0,769	0,867	2,814	2,697	00:00:24,830
ActiTraC^{MRA}	w25mcs100tf95	0,187	0,061	0,950	0,585	0,723	0,844	2,718	3,117	00:01:09,874
ActiTraC^{MRA}	w50mcs25tf95	0,691	0,113	0,950	0,700	0,796	0,877	2,425	2,462	00:00:09,907
ActiTraC^{MRA}	w50mcs50tf95	0,726	0,179	0,960	0,661	0,769	0,867	2,814	2,697	00:00:24,907
ActiTraC^{MRA}	w50mcs100tf95	0,187	0,061	0,950	0,585	0,723	0,844	2,718	3,117	00:01:10,561
ActiTraC^{MRA}	w100mcs25tf95	0,691	0,113	0,950	0,700	0,796	0,877	2,425	2,462	00:00:09,985
ActiTraC^{MRA}	w100mcs50tf95	0,726	0,179	0,960	0,661	0,769	0,867	2,814	2,697	00:00:25,405
ActiTraC^{MRA}	w100mcs100tf95	0,187	0,061	0,950	0,585	0,723	0,844	2,718	3,117	00:01:11,139
MR	Nominal-EuclideanDistance	0,000	0,000	0,720	0,564	0,619	0,669	3,024	2,825	00:00:04,356
MR	Nominal-FscoreSimilarity	0,000	0,000	0,793	0,630	0,689	0,742	2,838	2,830	00:00:03,919
MR	Numeric-EuclideanDistance	0,000	0,000	0,720	0,564	0,619	0,669	3,024	2,825	00:00:04,808
MR	Numeric-FscoreSimilarity	0,000	0,000	0,793	0,630	0,689	0,742	2,838	2,830	00:00:03,574
MRA	Nominal-EuclideanDistance	0,000	0,000	0,716	0,590	0,634	0,674	3,003	2,803	00:00:04,636
MRA	Nominal-FscoreSimilarity	0,000	0,000	0,712	0,568	0,613	0,658	3,032	2,819	00:00:04,136
MRA	Numeric-EuclideanDistance	0,000	0,000	0,716	0,590	0,634	0,674	3,003	2,803	00:00:04,543
MRA	Numeric-FscoreSimilarity	0,000	0,000	0,712	0,568	0,613	0,658	3,032	2,819	00:00:03,934
GED	-	0,000	0,000	0,796	0,718	0,740	0,766	2,828	2,830	00:00:05,651
LED	-	0,000	0,000	0,713	0,590	0,633	0,672	3,118	2,826	00:00:02,732
BOA	Nominal-EuclideanDistance	0,000	0,000	0,715	0,596	0,636	0,675	3,177	2,972	00:00:02,528
BOA	Nominal-FscoreSimilarity	0,000	0,000	0,715	0,594	0,635	0,673	3,171	2,972	00:00:02,607
BOA	Numeric-EuclideanDistance	0,000	0,000	0,715	0,596	0,636	0,675	3,177	2,972	00:00:02,561
BOA	Numeric-FscoreSimilarity	0,000	0,000	0,715	0,594	0,635	0,673	3,171	2,972	00:00:02,654
3-gram	Nominal-EuclideanDistance	0,000	0,000	0,860	0,537	0,656	0,762	3,290	3,399	00:00:02,735
3-gram	Nominal-FscoreSimilarity	0,000	0,000	0,868	0,534	0,660	0,770	3,098	3,197	00:00:02,541
3-gram	Numeric-EuclideanDistance	0,000	0,000	0,860	0,537	0,656	0,762	3,290	3,399	00:00:02,873
3-gram	Numeric-FscoreSimilarity	0,000	0,000	0,868	0,534	0,660	0,770	3,098	3,197	00:00:02,478
Markov	-	0,000	0,000	0,867	0,598	0,695	0,782	3,022	3,032	00:00:40,000
Random	-	0,008	0,003	0,831	0,485	0,607	0,721	3,060	3,061	00:00:01,225
No clustering	-	0,000	0,000	0,801	0,376	0,512	0,654	3,226	3,226	NA

Table D.4: Results of different trace clustering techniques with varying parameter settings for event log MCRM in case of the creation of 3 clusters.

Event log MCRM - 4 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,743	0,160	0,950	0,671	0,786	0,877	2,452	2,679	00:01:05,625
ActiTraC^{freq}	mcs50tf100	0,743	0,160	0,950	0,671	0,786	0,877	2,452	2,679	00:01:04,690
ActiTraC^{freq}	mcs100tf100	0,743	0,160	0,950	0,671	0,786	0,877	2,452	2,679	00:01:04,268
ActiTraC^{freq}	mcs25tf95	0,614	0,094	0,940	0,622	0,746	0,849	2,540	2,783	00:00:19,881
ActiTraC^{freq}	mcs50tf95	0,142	0,071	0,946	0,599	0,732	0,847	2,625	3,160	00:01:16,634
ActiTraC^{freq}	mcs100tf95	0,142	0,071	0,946	0,599	0,732	0,847	2,625	3,160	00:01:16,537
ActiTraC^{MRA}	w25mcs25tf100	0,738	0,175	0,951	0,757	0,824	0,886	2,301	2,180	00:00:11,752
ActiTraC^{MRA}	w25mcs50tf100	0,740	0,170	0,946	0,729	0,811	0,882	2,391	2,347	00:00:19,214
ActiTraC^{MRA}	w25mcs100tf100	0,745	0,165	0,938	0,618	0,745	0,850	2,469	2,769	00:01:03,466
ActiTraC^{MRA}	w50mcs25tf100	0,738	0,175	0,951	0,757	0,824	0,886	2,301	2,180	00:00:11,442
ActiTraC^{MRA}	w50mcs50tf100	0,740	0,170	0,946	0,729	0,811	0,882	2,391	2,347	00:00:18,571
ActiTraC^{MRA}	w50mcs100tf100	0,745	0,165	0,938	0,618	0,745	0,850	2,469	2,769	00:01:02,394
ActiTraC^{MRA}	w100mcs25tf100	0,738	0,175	0,951	0,757	0,824	0,886	2,301	2,180	00:00:11,558
ActiTraC^{MRA}	w100mcs50tf100	0,740	0,170	0,946	0,729	0,811	0,882	2,391	2,347	00:00:19,024
ActiTraC^{MRA}	w100mcs100tf100	0,745	0,165	0,938	0,618	0,745	0,850	2,469	2,769	00:01:02,935
ActiTraC^{MRA}	w25mcs25tf95	0,721	0,142	0,953	0,717	0,808	0,885	2,460	2,470	00:00:12,960
ActiTraC^{MRA}	w25mcs50tf95	0,726	0,179	0,960	0,669	0,775	0,870	2,667	2,684	00:00:25,547
ActiTraC^{MRA}	w25mcs100tf95	0,190	0,066	0,950	0,587	0,725	0,845	2,514	3,114	00:01:10,619
ActiTraC^{MRA}	w50mcs25tf95	0,721	0,142	0,953	0,717	0,808	0,885	2,460	2,470	00:00:12,696
ActiTraC^{MRA}	w50mcs50tf95	0,726	0,179	0,960	0,669	0,775	0,870	2,667	2,684	00:00:25,060
ActiTraC^{MRA}	w50mcs100tf95	0,190	0,066	0,950	0,587	0,725	0,845	2,514	3,114	00:01:10,950
ActiTraC^{MRA}	w100mcs25tf95	0,721	0,142	0,953	0,717	0,808	0,885	2,460	2,470	00:00:12,756
ActiTraC^{MRA}	w100mcs50tf95	0,726	0,179	0,960	0,669	0,775	0,870	2,667	2,684	00:00:25,325
ActiTraC^{MRA}	w100mcs100tf95	0,190	0,066	0,950	0,587	0,725	0,845	2,514	3,114	00:01:11,163
MR	Nominal-EuclideanDistance	0,000	0,000	0,713	0,558	0,611	0,660	2,886	2,773	00:00:04,559
MR	Nominal-FscoreSimilarity	0,000	0,000	0,788	0,619	0,677	0,731	2,840	2,792	00:00:03,574
MR	Numeric-EuclideanDistance	0,000	0,000	0,713	0,558	0,611	0,660	2,886	2,773	00:00:04,559
MR	Numeric-FscoreSimilarity	0,000	0,000	0,788	0,619	0,677	0,731	2,840	2,792	00:00:03,669
MRA	Nominal-EuclideanDistance	0,000	0,000	0,719	0,578	0,622	0,666	2,940	2,781	00:00:04,355
MRA	Nominal-FscoreSimilarity	0,000	0,000	0,784	0,588	0,648	0,711	2,962	2,796	00:00:03,403
MRA	Numeric-EuclideanDistance	0,000	0,000	0,719	0,578	0,622	0,666	2,940	2,781	00:00:04,262
MRA	Numeric-FscoreSimilarity	0,000	0,000	0,784	0,588	0,648	0,711	2,962	2,796	00:00:03,902
GED	-	0,000	0,000	0,802	0,697	0,718	0,752	2,830	2,793	00:00:05,589
LED	-	0,000	0,000	0,799	0,706	0,733	0,764	3,017	2,936	00:00:02,779
BOA	Nominal-EuclideanDistance	0,000	0,000	0,715	0,596	0,637	0,675	3,029	2,965	00:00:02,560
BOA	Nominal-FscoreSimilarity	0,000	0,000	0,800	0,724	0,741	0,766	2,963	2,790	00:00:02,419
BOA	Numeric-EuclideanDistance	0,000	0,000	0,715	0,596	0,637	0,675	3,029	2,965	00:00:02,561
BOA	Numeric-FscoreSimilarity	0,000	0,000	0,800	0,724	0,741	0,766	2,963	2,790	00:00:02,493
3-gram	Nominal-EuclideanDistance	0,000	0,000	0,862	0,554	0,672	0,772	3,131	3,348	00:00:02,992
3-gram	Nominal-FscoreSimilarity	0,132	0,066	0,902	0,553	0,684	0,799	2,873	3,119	00:00:02,541
3-gram	Numeric-EuclideanDistance	0,000	0,000	0,862	0,554	0,672	0,772	3,131	3,348	00:00:02,810
3-gram	Numeric-FscoreSimilarity	0,132	0,066	0,902	0,553	0,684	0,799	2,873	3,119	00:00:02,571
Markov	-	0,485	0,028	0,922	0,570	0,694	0,809	2,902	2,816	00:00:53,000
Random	-	0,002	0,002	0,823	0,474	0,592	0,707	2,997	2,972	00:00:01,245
No clustering	-	0,000	0,000	0,801	0,376	0,512	0,654	3,226	3,226	NA

Table D.5: Results of different trace clustering techniques with varying parameter settings for event log MCRM in case of the creation of 4 clusters.

Event log MCRM - 5 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,763	0,193	0,953	0,674	0,789	0,879	2,442	2,670	00:01:07,595
ActiTraC^{freq}	mcs50tf100	0,763	0,193	0,953	0,674	0,789	0,879	2,442	2,670	00:01:05,959
ActiTraC^{freq}	mcs100tf100	0,763	0,193	0,953	0,674	0,789	0,879	2,442	2,670	00:01:05,593
ActiTraC^{freq}	mcs25tf95	0,619	0,113	0,941	0,626	0,748	0,851	2,481	2,781	00:00:20,164
ActiTraC^{freq}	mcs50tf95	0,142	0,071	0,946	0,600	0,733	0,847	2,516	3,157	00:01:16,414
ActiTraC^{freq}	mcs100tf95	0,142	0,071	0,946	0,600	0,733	0,847	2,516	3,157	00:01:15,949
ActiTraC^{MRA}	w25mcs25tf100	0,755	0,193	0,961	0,782	0,842	0,901	2,361	2,201	00:00:14,409
ActiTraC^{MRA}	w25mcs50tf100	0,740	0,170	0,956	0,769	0,839	0,901	2,358	2,339	00:00:18,901
ActiTraC^{MRA}	w25mcs100tf100	0,745	0,165	0,950	0,656	0,776	0,872	2,420	2,735	00:01:02,845
ActiTraC^{MRA}	w50mcs25tf100	0,755	0,193	0,961	0,782	0,842	0,901	2,361	2,201	00:00:14,316
ActiTraC^{MRA}	w50mcs50tf100	0,740	0,170	0,956	0,769	0,839	0,901	2,358	2,339	00:00:19,047
ActiTraC^{MRA}	w50mcs100tf100	0,745	0,165	0,950	0,656	0,776	0,872	2,420	2,735	00:01:02,547
ActiTraC^{MRA}	w100mcs25tf100	0,755	0,193	0,961	0,782	0,842	0,901	2,361	2,201	00:00:14,111
ActiTraC^{MRA}	w100mcs50tf100	0,740	0,170	0,956	0,769	0,839	0,901	2,358	2,339	00:00:18,799
ActiTraC^{MRA}	w100mcs100tf100	0,745	0,165	0,950	0,656	0,776	0,872	2,420	2,735	00:01:02,433
ActiTraC^{MRA}	w25mcs25tf95	0,721	0,142	0,956	0,724	0,814	0,890	2,413	2,458	00:00:12,793
ActiTraC^{MRA}	w25mcs50tf95	0,732	0,184	0,962	0,677	0,782	0,875	2,648	2,673	00:00:25,959
ActiTraC^{MRA}	w25mcs100tf95	0,192	0,071	0,952	0,601	0,734	0,850	2,528	3,105	00:01:10,866
ActiTraC^{MRA}	w50mcs25tf95	0,721	0,142	0,956	0,724	0,814	0,890	2,413	2,458	00:00:12,792
ActiTraC^{MRA}	w50mcs50tf95	0,732	0,184	0,962	0,677	0,782	0,875	2,648	2,673	00:00:25,979
ActiTraC^{MRA}	w50mcs100tf95	0,192	0,071	0,952	0,601	0,734	0,850	2,528	3,105	00:01:10,942
ActiTraC^{MRA}	w100mcs25tf95	0,721	0,142	0,956	0,724	0,814	0,890	2,413	2,458	00:00:12,724
ActiTraC^{MRA}	w100mcs50tf95	0,732	0,184	0,962	0,677	0,782	0,875	2,648	2,673	00:00:25,791
ActiTraC^{MRA}	w100mcs100tf95	0,192	0,071	0,952	0,601	0,734	0,850	2,528	3,105	00:01:11,304
MR	Nominal-EuclideanDistance	0,000	0,000	0,786	0,688	0,711	0,743	2,837	2,756	00:00:04,543
MR	Nominal-FscoreSimilarity	0,000	0,000	0,796	0,623	0,683	0,737	2,772	2,768	00:00:03,746
MR	Numeric-EuclideanDistance	0,000	0,000	0,786	0,688	0,711	0,743	2,837	2,756	00:00:04,542
MR	Numeric-FscoreSimilarity	0,000	0,000	0,796	0,623	0,683	0,737	2,772	2,768	00:00:03,653
MRA	Nominal-EuclideanDistance	0,000	0,000	0,803	0,637	0,690	0,742	2,905	2,953	00:00:04,121
MRA	Nominal-FscoreSimilarity	0,000	0,000	0,790	0,590	0,651	0,716	2,904	2,764	00:00:03,372
MRA	Numeric-EuclideanDistance	0,000	0,000	0,803	0,637	0,690	0,742	2,905	2,953	00:00:04,480
MRA	Numeric-FscoreSimilarity	0,000	0,000	0,790	0,590	0,651	0,716	2,904	2,764	00:00:03,887
GED	-	0,000	0,000	0,807	0,703	0,726	0,759	2,920	2,796	00:00:05,542
LED	-	0,000	0,000	0,806	0,690	0,718	0,754	2,896	2,901	00:00:02,716
BOA	Nominal-EuclideanDistance	0,000	0,000	0,800	0,718	0,739	0,766	2,921	2,895	00:00:02,513
BOA	Nominal-FscoreSimilarity	0,030	0,028	0,803	0,717	0,735	0,763	2,873	2,775	00:00:02,560
BOA	Numeric-EuclideanDistance	0,000	0,000	0,800	0,718	0,739	0,766	2,921	2,895	00:00:02,654
BOA	Numeric-FscoreSimilarity	0,030	0,028	0,803	0,717	0,735	0,763	2,873	2,775	00:00:02,364
3-gram	Nominal-EuclideanDistance	0,000	0,000	0,860	0,595	0,699	0,785	3,026	3,256	00:00:02,912
3-gram	Nominal-FscoreSimilarity	0,132	0,066	0,926	0,475	0,626	0,776	2,904	3,213	00:00:02,510
3-gram	Numeric-EuclideanDistance	0,000	0,000	0,860	0,595	0,699	0,785	3,026	3,256	00:00:02,967
3-gram	Numeric-FscoreSimilarity	0,132	0,066	0,926	0,475	0,626	0,776	2,904	3,213	00:00:02,477
Markov	-	0,009	0,009	0,832	0,536	0,637	0,734	2,991	2,812	00:01:12,000
Random	-	0,096	0,005	0,819	0,461	0,583	0,701	2,897	2,929	00:00:01,183
No clustering	-	0,000	0,000	0,801	0,376	0,512	0,654	3,226	3,226	NA

Table D.6: Results of different trace clustering techniques with varying parameter settings for event log MCRM in case of the creation of 5 clusters.

Event log TSL - 3 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,880	0,260	0,969	0,898	0,931	0,953	3,404	3,440	00:04:15,945
ActiTraC^{freq}	mcs50tf100	0,881	0,257	0,969	0,904	0,935	0,955	3,332	3,423	00:13:12,506
ActiTraC^{freq}	mcs100tf100	0,878	0,232	0,967	0,894	0,929	0,952	3,255	3,508	01:47:02,954
ActiTraC^{freq}	mcs25tf95	0,786	0,181	0,956	0,891	0,921	0,941	3,751	4,085	00:07:22,314
ActiTraC^{freq}	mcs50tf95	0,785	0,176	0,956	0,891	0,921	0,941	3,747	4,086	00:11:36,337
ActiTraC^{freq}	mcs100tf95	0,055	0,037	0,946	0,782	0,856	0,908	3,601	4,136	01:54:45,520
ActiTraC^{MRA}	w25mcs25tf100	0,821	0,318	0,949	0,843	0,892	0,925	3,061	3,145	00:00:31,624
ActiTraC^{MRA}	w25mcs50tf100	0,833	0,195	0,954	0,836	0,890	0,927	3,323	3,551	00:20:26,875
ActiTraC^{MRA}	w25mcs100tf100	0,839	0,202	0,954	0,828	0,886	0,926	3,379	3,441	01:29:30,551
ActiTraC^{MRA}	w50mcs25tf100	0,749	0,248	0,929	0,803	0,859	0,899	3,279	3,274	00:09:40,142
ActiTraC^{MRA}	w50mcs50tf100	0,833	0,195	0,954	0,836	0,890	0,927	3,323	3,551	00:20:09,719
ActiTraC^{MRA}	w50mcs100tf100	0,839	0,202	0,954	0,828	0,886	0,926	3,379	3,441	01:28:28,770
ActiTraC^{MRA}	w100mcs25tf100	0,749	0,248	0,929	0,803	0,859	0,899	3,279	3,274	00:09:39,344
ActiTraC^{MRA}	w100mcs50tf100	0,833	0,195	0,954	0,836	0,890	0,927	3,323	3,551	00:20:11,454
ActiTraC^{MRA}	w100mcs100tf100	0,839	0,202	0,954	0,828	0,886	0,926	3,379	3,441	01:28:30,583
ActiTraC^{MRA}	w25mcs25tf95	0,425	0,072	0,934	0,769	0,841	0,894	3,750	3,927	00:04:51,664
ActiTraC^{MRA}	w25mcs50tf95	0,402	0,082	0,948	0,831	0,885	0,922	3,657	3,749	00:09:48,188
ActiTraC^{MRA}	w25mcs100tf95	0,372	0,077	0,942	0,794	0,861	0,908	3,731	3,997	01:40:37,355
ActiTraC^{MRA}	w50mcs25tf95	0,425	0,072	0,934	0,769	0,841	0,894	3,750	3,927	00:04:46,961
ActiTraC^{MRA}	w50mcs50tf95	0,402	0,082	0,948	0,831	0,885	0,922	3,657	3,749	00:09:40,203
ActiTraC^{MRA}	w50mcs100tf95	0,372	0,078	0,943	0,794	0,861	0,908	3,731	3,997	01:40:27,659
ActiTraC^{MRA}	w100mcs25tf95	0,425	0,072	0,934	0,769	0,841	0,894	3,750	3,927	00:04:47,367
ActiTraC^{MRA}	w100mcs50tf95	0,402	0,082	0,948	0,831	0,885	0,922	3,657	3,749	00:09:40,001
ActiTraC^{MRA}	w100mcs100tf95	0,372	0,078	0,943	0,794	0,861	0,908	3,731	3,997	01:40:30,502
MR	Nominal-EuclideanDistance	0,029	0,012	0,731	0,794	0,742	0,727	4,239	4,230	00:05:23,069
MR	Nominal-FscoreSimilarity	0,037	0,009	0,696	0,740	0,690	0,682	4,290	4,290	00:01:52,132
MR	Numeric-EuclideanDistance	0,029	0,012	0,731	0,794	0,742	0,727	4,239	4,230	00:05:22,772
MR	Numeric-FscoreSimilarity	0,037	0,009	0,696	0,740	0,690	0,682	4,290	4,290	00:01:58,601
MRA	Nominal-EuclideanDistance	0,039	0,013	0,829	0,725	0,771	0,803	3,915	4,556	00:03:17,857
MRA	Nominal-FscoreSimilarity	0,398	0,024	0,793	0,750	0,768	0,782	4,535	4,820	00:01:58,146
MRA	Numeric-EuclideanDistance	0,039	0,013	0,829	0,725	0,771	0,803	3,915	4,556	00:03:14,172
MRA	Numeric-FscoreSimilarity	0,398	0,024	0,793	0,750	0,768	0,782	4,535	4,820	00:02:04,288
GED	-	0,032	0,019	0,732	0,727	0,712	0,717	4,298	4,162	00:02:14,364
LED	-	0,066	0,015	0,742	0,604	0,659	0,703	4,412	4,161	00:01:58,772
BOA	Nominal-EuclideanDistance	0,035	0,007	0,767	0,595	0,664	0,719	4,412	4,447	00:01:38,274
BOA	Nominal-FscoreSimilarity	0,051	0,021	0,726	0,534	0,605	0,668	4,327	4,291	00:01:20,259
BOA	Numeric-EuclideanDistance	0,035	0,007	0,767	0,595	0,664	0,719	4,412	4,447	00:01:48,898
BOA	Numeric-FscoreSimilarity	0,051	0,021	0,726	0,534	0,605	0,668	4,327	4,291	00:01:10,411
3-gram	Nominal-EuclideanDistance	0,039	0,016	0,746	0,854	0,795	0,764	3,667	4,355	00:04:34,151
3-gram	Nominal-FscoreSimilarity	0,038	0,014	0,696	0,785	0,735	0,710	4,224	4,153	00:02:23,981
3-gram	Numeric-EuclideanDistance	0,039	0,016	0,746	0,854	0,795	0,764	3,667	4,355	00:04:34,052
3-gram	Numeric-FscoreSimilarity	0,038	0,014	0,696	0,785	0,735	0,710	4,224	4,153	00:02:21,077
Markov	-	0,111	0,013	0,818	0,695	0,751	0,790	4,148	4,124	03:59:18,000
Random	-	0,093	0,013	0,725	0,596	0,650	0,692	4,035	4,034	00:00:22,746
No clustering	-	0,037	0,009	0,745	0,862	0,799	0,766	4,368	4,368	NA

Table D.7: Results of different trace clustering techniques with varying parameter settings for event log TSL in case of the creation of 3 clusters.

Event log TSL - 4 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_{A}	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_{A}$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,889	0,271	0,971	0,901	0,933	0,955	3,338	3,435	00:05:13,258
ActiTraC^{freq}	mcs50tf100	0,890	0,273	0,972	0,907	0,937	0,957	3,301	3,420	00:13:57,369
ActiTraC^{freq}	mcs100tf100	0,889	0,251	0,970	0,895	0,931	0,954	3,227	3,503	01:46:44,227
ActiTraC^{freq}	mcs25tf95	0,788	0,188	0,957	0,892	0,922	0,942	3,578	4,076	00:07:55,394
ActiTraC^{freq}	mcs50tf95	0,787	0,182	0,957	0,891	0,921	0,942	3,594	4,078	00:12:04,481
ActiTraC^{freq}	mcs100tf95	0,056	0,042	0,946	0,782	0,856	0,908	3,425	4,131	01:53:03,436
ActiTraC^{MRA}	w25mcs25tf100	0,849	0,309	0,960	0,863	0,907	0,937	3,106	3,149	00:04:36,821
ActiTraC^{MRA}	w25mcs50tf100	0,848	0,212	0,958	0,843	0,896	0,932	3,362	3,551	00:21:57,379
ActiTraC^{MRA}	w25mcs100tf100	0,859	0,221	0,960	0,835	0,893	0,932	3,301	3,430	01:30:08,754
ActiTraC^{MRA}	w50mcs25tf100	0,783	0,257	0,940	0,799	0,860	0,905	3,283	3,276	00:13:21,746
ActiTraC^{MRA}	w50mcs50tf100	0,848	0,212	0,958	0,843	0,896	0,932	3,362	3,551	00:21:44,170
ActiTraC^{MRA}	w50mcs100tf100	0,859	0,221	0,960	0,835	0,893	0,932	3,301	3,430	01:30:05,868
ActiTraC^{MRA}	w100mcs25tf100	0,783	0,257	0,940	0,799	0,860	0,905	3,283	3,276	00:13:21,964
ActiTraC^{MRA}	w100mcs50tf100	0,848	0,212	0,958	0,843	0,896	0,932	3,362	3,551	00:21:43,482
ActiTraC^{MRA}	w100mcs100tf100	0,859	0,221	0,960	0,835	0,893	0,932	3,301	3,430	01:30:05,858
ActiTraC^{MRA}	w25mcs25tf95	0,454	0,112	0,945	0,788	0,857	0,907	3,632	3,885	00:06:30,693
ActiTraC^{MRA}	w25mcs50tf95	0,403	0,087	0,954	0,832	0,887	0,925	3,611	3,740	00:10:49,350
ActiTraC^{MRA}	w25mcs100tf95	0,374	0,084	0,944	0,794	0,861	0,908	3,590	3,992	01:40:46,079
ActiTraC^{MRA}	w50mcs25tf95	0,453	0,108	0,945	0,793	0,860	0,908	3,673	3,897	00:06:41,427
ActiTraC^{MRA}	w50mcs50tf95	0,403	0,087	0,954	0,832	0,887	0,925	3,611	3,740	00:10:50,032
ActiTraC^{MRA}	w50mcs100tf95	0,374	0,084	0,944	0,794	0,861	0,908	3,590	3,992	01:40:45,126
ActiTraC^{MRA}	w100mcs25tf95	0,453	0,108	0,945	0,793	0,860	0,908	3,673	3,897	00:06:41,068
ActiTraC^{MRA}	w100mcs50tf95	0,403	0,087	0,954	0,832	0,887	0,925	3,611	3,740	00:10:49,890
ActiTraC^{MRA}	w100mcs100tf95	0,374	0,084	0,944	0,794	0,861	0,908	3,590	3,992	01:40:43,470
MR	Nominal-EuclideanDistance	0,039	0,018	0,725	0,740	0,713	0,711	4,212	4,234	00:05:10,851
MR	Nominal-FscoreSimilarity	0,030	0,009	0,691	0,683	0,661	0,668	4,211	4,110	00:01:59,069
MR	Numeric-EuclideanDistance	0,039	0,018	0,725	0,740	0,713	0,711	4,212	4,234	00:05:14,382
MR	Numeric-FscoreSimilarity	0,030	0,009	0,691	0,683	0,661	0,668	4,211	4,110	00:01:57,351
MRA	Nominal-EuclideanDistance	0,449	0,030	0,818	0,751	0,778	0,800	4,061	4,634	00:03:10,842
MRA	Nominal-FscoreSimilarity	0,398	0,024	0,792	0,743	0,762	0,778	4,497	4,814	00:02:03,287
MRA	Numeric-EuclideanDistance	0,449	0,030	0,818	0,751	0,778	0,800	4,061	4,634	00:03:08,845
MRA	Numeric-FscoreSimilarity	0,398	0,024	0,792	0,743	0,762	0,778	4,497	4,814	00:02:05,975
GED	-	0,042	0,019	0,738	0,746	0,721	0,721	4,126	4,055	00:02:18,849
LED	-	0,068	0,015	0,763	0,636	0,686	0,725	4,164	4,273	00:01:59,631
BOA	Nominal-EuclideanDistance	0,104	0,022	0,773	0,561	0,643	0,711	4,318	4,304	00:01:43,867
BOA	Nominal-FscoreSimilarity	0,050	0,020	0,724	0,527	0,598	0,663	4,300	4,313	00:01:28,337
BOA	Numeric-EuclideanDistance	0,104	0,022	0,773	0,561	0,643	0,711	4,318	4,304	00:01:49,273
BOA	Numeric-FscoreSimilarity	0,050	0,020	0,724	0,527	0,598	0,663	4,300	4,313	00:01:22,888
3-gram	Nominal-EuclideanDistance	0,040	0,020	0,730	0,808	0,761	0,740	3,915	4,392	00:04:36,825
3-gram	Nominal-FscoreSimilarity	0,038	0,010	0,689	0,630	0,651	0,671	4,206	4,298	00:02:21,842
3-gram	Numeric-EuclideanDistance	0,040	0,020	0,730	0,808	0,761	0,740	3,915	4,392	00:04:31,477
3-gram	Numeric-FscoreSimilarity	0,038	0,010	0,689	0,630	0,651	0,671	4,206	4,298	00:02:24,340
Markov	-	0,129	0,025	0,820	0,607	0,690	0,760	4,308	4,348	05:16:54,000
Random	-	0,132	0,015	0,767	0,619	0,683	0,731	4,004	4,007	00:00:22,966
No clustering	-	0,037	0,009	0,745	0,862	0,799	0,766	4,368	4,368	NA

Table D.8: Results of different trace clustering techniques with varying parameter settings for event log TSL in case of the creation of 4 clusters.

Event log TSL - 5 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,894	0,281	0,972	0,902	0,934	0,956	3,296	3,433	00:05:49,320
ActiTraC^{freq}	mcs50tf100	0,895	0,284	0,973	0,907	0,938	0,958	3,267	3,417	00:14:32,306
ActiTraC^{freq}	mcs100tf100	0,895	0,269	0,972	0,896	0,932	0,955	3,248	3,502	01:47:10,398
ActiTraC^{freq}	mcs25tf95	0,791	0,197	0,958	0,893	0,923	0,943	3,493	4,069	00:08:20,909
ActiTraC^{freq}	mcs50tf95	0,790	0,191	0,959	0,892	0,922	0,943	3,503	4,071	00:12:30,106
ActiTraC^{freq}	mcs100tf95	0,057	0,047	0,947	0,783	0,856	0,908	3,288	4,127	01:53:04,998
ActiTraC^{MRA}	w25mcs25tf100	0,878	0,327	0,968	0,873	0,916	0,946	3,129	3,154	00:09:41,141
ActiTraC^{MRA}	w25mcs50tf100	0,860	0,231	0,962	0,846	0,899	0,935	3,421	3,554	00:22:40,387
ActiTraC^{MRA}	w25mcs100tf100	0,874	0,246	0,966	0,841	0,898	0,937	3,282	3,426	01:31:16,577
ActiTraC^{MRA}	w50mcs25tf100	0,837	0,278	0,957	0,815	0,877	0,922	3,310	3,290	00:17:04,771
ActiTraC^{MRA}	w50mcs50tf100	0,860	0,231	0,962	0,846	0,899	0,935	3,421	3,554	00:22:41,324
ActiTraC^{MRA}	w50mcs100tf100	0,874	0,246	0,966	0,841	0,898	0,937	3,282	3,426	01:31:14,891
ActiTraC^{MRA}	w100mcs25tf100	0,837	0,278	0,957	0,815	0,877	0,922	3,310	3,290	00:17:04,599
ActiTraC^{MRA}	w100mcs50tf100	0,860	0,231	0,962	0,846	0,899	0,935	3,421	3,554	00:22:41,043
ActiTraC^{MRA}	w100mcs100tf100	0,874	0,246	0,966	0,841	0,898	0,937	3,282	3,426	01:31:18,757
ActiTraC^{MRA}	w25mcs25tf95	0,458	0,124	0,947	0,792	0,860	0,909	3,591	3,881	00:07:06,677
ActiTraC^{MRA}	w25mcs50tf95	0,412	0,101	0,957	0,831	0,886	0,926	3,553	3,728	00:11:55,981
ActiTraC^{MRA}	w25mcs100tf95	0,376	0,090	0,945	0,796	0,862	0,909	3,532	3,988	01:41:08,298
ActiTraC^{MRA}	w50mcs25tf95	0,457	0,121	0,950	0,806	0,870	0,915	3,624	3,886	00:07:24,067
ActiTraC^{MRA}	w50mcs50tf95	0,412	0,101	0,957	0,831	0,886	0,926	3,553	3,728	00:11:56,794
ActiTraC^{MRA}	w50mcs100tf95	0,376	0,090	0,945	0,795	0,861	0,909	3,534	3,989	01:41:07,439
ActiTraC^{MRA}	w100mcs25tf95	0,457	0,121	0,950	0,806	0,870	0,915	3,624	3,886	00:07:24,114
ActiTraC^{MRA}	w100mcs50tf95	0,412	0,101	0,957	0,831	0,886	0,926	3,553	3,728	00:11:57,169
ActiTraC^{MRA}	w100mcs100tf95	0,376	0,090	0,945	0,795	0,861	0,909	3,534	3,989	01:41:11,594
MR	Nominal-EuclideanDistance	0,039	0,021	0,725	0,736	0,709	0,709	4,022	4,260	00:05:32,381
MR	Nominal-FscoreSimilarity	0,040	0,016	0,713	0,544	0,605	0,659	4,110	4,041	00:01:54,179
MR	Numeric-EuclideanDistance	0,039	0,021	0,725	0,736	0,709	0,709	4,022	4,260	00:05:23,460
MR	Numeric-FscoreSimilarity	0,040	0,016	0,713	0,544	0,605	0,659	4,110	4,041	00:01:50,788
MRA	Nominal-EuclideanDistance	0,449	0,030	0,820	0,745	0,773	0,798	4,013	4,656	00:03:14,500
MRA	Nominal-FscoreSimilarity	0,407	0,038	0,815	0,739	0,769	0,794	4,522	4,838	00:01:55,757
MRA	Numeric-EuclideanDistance	0,449	0,030	0,820	0,745	0,773	0,798	4,013	4,656	00:03:10,485
MRA	Numeric-FscoreSimilarity	0,407	0,038	0,815	0,739	0,769	0,794	4,522	4,838	00:02:04,662
GED	-	0,042	0,019	0,750	0,643	0,677	0,711	4,045	4,083	00:02:17,974
LED	-	0,075	0,022	0,764	0,588	0,655	0,711	4,193	4,350	00:01:59,899
BOA	Nominal-EuclideanDistance	0,097	0,019	0,778	0,547	0,632	0,707	4,248	4,318	00:01:44,054
BOA	Nominal-FscoreSimilarity	0,051	0,024	0,744	0,526	0,600	0,671	4,310	4,273	00:01:30,727
BOA	Numeric-EuclideanDistance	0,097	0,019	0,778	0,547	0,632	0,707	4,248	4,318	00:01:49,429
BOA	Numeric-FscoreSimilarity	0,051	0,024	0,744	0,526	0,600	0,671	4,310	4,273	00:01:23,686
3-gram	Nominal-EuclideanDistance	0,040	0,020	0,722	0,784	0,739	0,724	3,978	4,371	00:04:33,364
3-gram	Nominal-FscoreSimilarity	0,041	0,015	0,695	0,623	0,650	0,674	4,114	4,137	00:02:20,000
3-gram	Numeric-EuclideanDistance	0,040	0,020	0,722	0,784	0,739	0,724	3,978	4,371	00:04:38,705
3-gram	Numeric-FscoreSimilarity	0,041	0,015	0,695	0,623	0,650	0,674	4,114	4,137	00:02:17,722
Markov	-	0,222	0,045	0,831	0,709	0,764	0,803	3,912	3,883	05:34:30,000
Random	-	0,175	0,016	0,773	0,615	0,681	0,732	4,074	4,046	00:00:26,272
No clustering	-	0,037	0,009	0,745	0,862	0,799	0,766	4,368	4,368	NA

Table D.9: Results of different trace clustering techniques with varying parameter settings for event log TSL in case of the creation of 5 clusters.

Event log ICP - 3 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_{A}	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_{A}$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,829	0,304	0,961	0,812	0,865	0,911	3,638	2,750	00:03:32,119
ActiTraC^{freq}	mcs50tf100	0,862	0,240	0,966	0,509	0,662	0,813	4,136	4,128	00:47:53,592
ActiTraC^{freq}	mcs100tf100	0,862	0,240	0,966	0,509	0,662	0,813	4,136	4,128	00:47:55,451
ActiTraC^{freq}	mcs25tf95	0,760	0,257	0,969	0,818	0,873	0,923	3,376	2,758	00:00:42,890
ActiTraC^{freq}	mcs50tf95	0,159	0,063	0,946	0,339	0,497	0,692	4,590	5,053	00:12:09,653
ActiTraC^{freq}	mcs100tf95	0,170	0,055	0,947	0,318	0,476	0,679	4,423	5,408	00:49:43,549
ActiTraC^{MRA}	w25mcs25tf100	0,793	0,265	0,949	0,735	0,791	0,858	3,910	3,207	00:07:32,160
ActiTraC^{MRA}	w25mcs50tf100	0,770	0,223	0,943	0,633	0,730	0,827	4,172	3,717	00:07:47,722
ActiTraC^{MRA}	w25mcs100tf100	0,803	0,216	0,952	0,474	0,631	0,789	4,091	4,493	00:43:44,261
ActiTraC^{MRA}	w50mcs25tf100	0,781	0,315	0,947	0,740	0,810	0,874	2,904	2,717	00:04:09,619
ActiTraC^{MRA}	w50mcs50tf100	0,740	0,177	0,935	0,512	0,654	0,792	3,862	3,830	00:16:46,802
ActiTraC^{MRA}	w50mcs100tf100	0,792	0,192	0,946	0,509	0,659	0,804	3,768	3,889	00:36:41,382
ActiTraC^{MRA}	w100mcs25tf100	0,781	0,315	0,947	0,742	0,812	0,876	2,889	2,708	00:04:14,681
ActiTraC^{MRA}	w100mcs50tf100	0,740	0,177	0,935	0,514	0,656	0,793	3,841	3,819	00:16:36,147
ActiTraC^{MRA}	w100mcs100tf100	0,792	0,192	0,946	0,509	0,659	0,804	3,768	3,889	00:36:20,195
ActiTraC^{MRA}	w25mcs25tf95	0,017	0,028	0,953	0,361	0,523	0,717	5,137	5,301	00:12:11,825
ActiTraC^{MRA}	w25mcs50tf95	0,052	0,043	0,958	0,365	0,528	0,721	5,164	5,285	00:17:44,223
ActiTraC^{MRA}	w25mcs100tf95	0,019	0,012	0,953	0,329	0,487	0,687	5,189	5,740	00:45:38,836
ActiTraC^{MRA}	w50mcs25tf95	0,567	0,060	0,931	0,446	0,594	0,752	5,964	6,315	00:08:23,034
ActiTraC^{MRA}	w50mcs50tf95	0,610	0,033	0,949	0,482	0,614	0,757	5,915	7,319	00:14:09,541
ActiTraC^{MRA}	w50mcs100tf95	0,032	0,032	0,957	0,331	0,490	0,690	6,019	7,256	00:49:33,285
ActiTraC^{MRA}	w100mcs25tf95	0,542	0,049	0,934	0,449	0,596	0,754	5,898	6,270	00:08:28,268
ActiTraC^{MRA}	w100mcs50tf95	0,606	0,024	0,946	0,480	0,615	0,760	5,889	7,367	00:13:48,760
ActiTraC^{MRA}	w100mcs100tf95	0,083	0,045	0,955	0,335	0,495	0,696	4,916	5,975	00:48:36,910
MR	Nominal-EuclideanDistance	0,551	0,055	0,870	0,515	0,622	0,734	6,224	5,966	00:03:01,477
MR	Nominal-FscoreSimilarity	0,392	0,042	0,846	0,550	0,645	0,742	6,085	5,940	00:02:23,062
MR	Numeric-EuclideanDistance	0,551	0,055	0,870	0,515	0,622	0,734	6,224	5,966	00:03:39,160
MR	Numeric-FscoreSimilarity	0,392	0,042	0,846	0,550	0,645	0,742	6,085	5,940	00:02:01,011
MRA	Nominal-EuclideanDistance	0,036	0,014	0,751	0,281	0,408	0,561	6,331	6,141	00:03:21,195
MRA	Nominal-FscoreSimilarity	0,033	0,015	0,793	0,288	0,420	0,583	7,009	6,727	00:02:22,236
MRA	Numeric-EuclideanDistance	0,036	0,014	0,751	0,281	0,408	0,561	6,331	6,141	00:03:00,482
MRA	Numeric-FscoreSimilarity	0,033	0,015	0,793	0,288	0,420	0,583	7,009	6,727	00:02:16,401
GED	-	0,173	0,024	0,736	0,246	0,367	0,522	5,532	6,552	00:01:41,938
LED	-	0,147	0,038	0,710	0,253	0,360	0,501	5,874	6,071	00:02:08,769
BOA	Nominal-EuclideanDistance	0,003	0,007	0,730	0,253	0,372	0,524	6,082	6,954	00:01:12,025
BOA	Nominal-FscoreSimilarity	0,002	0,006	0,737	0,276	0,400	0,550	5,755	6,502	00:01:14,851
BOA	Numeric-EuclideanDistance	0,003	0,007	0,730	0,253	0,372	0,524	6,082	6,954	00:01:12,758
BOA	Numeric-FscoreSimilarity	0,002	0,006	0,737	0,276	0,400	0,550	5,755	6,502	00:01:14,538
3-gram	Nominal-EuclideanDistance	0,025	0,013	0,759	0,335	0,446	0,582	5,531	5,772	00:03:45,207
3-gram	Nominal-FscoreSimilarity	0,048	0,037	0,773	0,405	0,518	0,640	5,650	5,976	00:02:51,303
3-gram	Numeric-EuclideanDistance	0,025	0,013	0,759	0,335	0,446	0,582	5,531	5,772	00:03:52,811
3-gram	Numeric-FscoreSimilarity	0,048	0,037	0,773	0,405	0,518	0,640	5,650	5,976	00:02:33,086
Markov	-	0,009	0,010	0,857	0,341	0,487	0,656	5,522	5,596	01:17:50,000
Random	-	0,007	0,005	0,740	0,257	0,379	0,533	6,497	6,484	00:00:17,468
No clustering	-	0,003	0,006	0,693	0,237	0,353	0,500	7,252	7,252	NA

Table D.10: Results of different trace clustering techniques with varying parameter settings for event log ICP in case of the creation of 3 clusters.

Event log ICP - 4 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_A	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_A$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,852	0,325	0,969	0,804	0,851	0,902	3,804	2,853	00:06:01,803
ActiTraC^{freq}	mcs50tf100	0,876	0,277	0,970	0,522	0,674	0,822	4,026	4,119	00:49:55,948
ActiTraC^{freq}	mcs100tf100	0,876	0,277	0,970	0,522	0,674	0,822	4,026	4,119	00:49:52,058
ActiTraC^{freq}	mcs25tf95	0,772	0,262	0,974	0,810	0,864	0,917	3,698	2,866	00:03:00,308
ActiTraC^{freq}	mcs50tf95	0,164	0,084	0,948	0,342	0,500	0,696	4,591	5,051	00:12:54,012
ActiTraC^{freq}	mcs100tf95	0,174	0,073	0,948	0,320	0,478	0,681	4,262	5,404	00:51:10,125
ActiTraC^{MRA}	w25mcs25tf100	0,828	0,297	0,958	0,731	0,792	0,864	3,843	3,203	00:09:41,532
ActiTraC^{MRA}	w25mcs50tf100	0,807	0,263	0,954	0,659	0,752	0,844	4,103	3,720	00:09:27,517
ActiTraC^{MRA}	w25mcs100tf100	0,830	0,257	0,960	0,494	0,649	0,803	4,102	4,480	00:46:03,038
ActiTraC^{MRA}	w50mcs25tf100	0,799	0,332	0,951	0,748	0,811	0,876	3,108	2,831	00:07:22,457
ActiTraC^{MRA}	w50mcs50tf100	0,791	0,227	0,947	0,544	0,683	0,815	3,806	3,816	00:18:08,472
ActiTraC^{MRA}	w50mcs100tf100	0,807	0,221	0,953	0,528	0,678	0,818	3,763	3,886	00:37:14,662
ActiTraC^{MRA}	w100mcs25tf100	0,799	0,332	0,951	0,749	0,813	0,877	3,096	2,824	00:07:23,270
ActiTraC^{MRA}	w100mcs50tf100	0,791	0,229	0,947	0,546	0,685	0,816	3,791	3,807	00:17:55,316
ActiTraC^{MRA}	w100mcs100tf100	0,807	0,221	0,953	0,528	0,678	0,818	3,763	3,886	00:36:56,584
ActiTraC^{MRA}	w25mcs25tf95	0,030	0,035	0,960	0,379	0,543	0,733	4,910	5,236	00:13:27,260
ActiTraC^{MRA}	w25mcs50tf95	0,057	0,052	0,960	0,371	0,534	0,727	4,742	5,229	00:18:38,534
ActiTraC^{MRA}	w25mcs100tf95	0,022	0,023	0,955	0,332	0,490	0,690	4,756	5,708	00:47:05,428
ActiTraC^{MRA}	w50mcs25tf95	0,598	0,082	0,950	0,470	0,619	0,776	5,780	6,277	00:09:29,767
ActiTraC^{MRA}	w50mcs50tf95	0,638	0,065	0,955	0,502	0,630	0,769	5,576	7,215	00:14:35,353
ActiTraC^{MRA}	w50mcs100tf95	0,039	0,043	0,959	0,335	0,494	0,694	5,435	7,127	00:49:49,283
ActiTraC^{MRA}	w100mcs25tf95	0,571	0,081	0,949	0,467	0,615	0,772	5,434	6,097	00:09:34,595
ActiTraC^{MRA}	w100mcs50tf95	0,619	0,038	0,952	0,509	0,639	0,776	5,438	7,245	00:14:10,088
ActiTraC^{MRA}	w100mcs100tf95	0,097	0,064	0,957	0,339	0,500	0,701	4,643	5,940	00:48:44,097
MR	Nominal-EuclideanDistance	0,562	0,067	0,893	0,524	0,636	0,752	6,185	5,852	00:03:23,119
MR	Nominal-FscoreSimilarity	0,395	0,046	0,860	0,553	0,650	0,750	5,740	5,589	00:02:28,917
MR	Numeric-EuclideanDistance	0,562	0,067	0,893	0,524	0,636	0,752	6,185	5,852	00:03:47,906
MR	Numeric-FscoreSimilarity	0,395	0,046	0,860	0,553	0,650	0,750	5,740	5,589	00:02:18,628
MRA	Nominal-EuclideanDistance	0,037	0,017	0,748	0,283	0,409	0,561	6,372	6,175	00:03:15,260
MRA	Nominal-FscoreSimilarity	0,051	0,049	0,795	0,313	0,446	0,603	7,363	6,965	00:02:21,596
MRA	Numeric-EuclideanDistance	0,037	0,017	0,748	0,283	0,409	0,561	6,372	6,175	00:03:10,800
MRA	Numeric-FscoreSimilarity	0,051	0,049	0,795	0,313	0,446	0,603	7,363	6,965	00:02:14,345
GED	-	0,179	0,023	0,836	0,288	0,425	0,600	5,606	6,184	00:01:46,563
LED	-	0,151	0,026	0,847	0,303	0,433	0,602	6,133	6,203	00:02:11,296
BOA	Nominal-EuclideanDistance	0,018	0,009	0,736	0,370	0,485	0,604	5,806	6,689	00:01:22,225
BOA	Nominal-FscoreSimilarity	0,003	0,007	0,773	0,268	0,394	0,554	5,853	6,273	00:01:17,662
BOA	Numeric-EuclideanDistance	0,018	0,009	0,736	0,370	0,485	0,604	5,806	6,689	00:01:22,881
BOA	Numeric-FscoreSimilarity	0,003	0,007	0,773	0,268	0,394	0,554	5,853	6,273	00:01:18,038
3-gram	Nominal-EuclideanDistance	0,029	0,030	0,821	0,448	0,553	0,672	5,142	5,573	00:03:51,285
3-gram	Nominal-FscoreSimilarity	0,199	0,069	0,805	0,509	0,611	0,707	5,162	5,492	00:02:40,789
3-gram	Numeric-EuclideanDistance	0,029	0,030	0,821	0,448	0,553	0,672	5,142	5,573	00:03:49,683
3-gram	Numeric-FscoreSimilarity	0,199	0,069	0,805	0,509	0,611	0,707	5,162	5,492	00:02:32,133
Markov	-	0,461	0,031	0,883	0,530	0,642	0,754	5,513	5,664	01:42:56,000
Random	-	0,078	0,009	0,811	0,304	0,438	0,601	6,606	6,552	00:00:16,546
No clustering	-	0,003	0,006	0,693	0,237	0,353	0,500	7,252	7,252	NA

Table D.11: Results of different trace clustering techniques with varying parameter settings for event log ICP in case of the creation of 4 clusters.

Event log ICP - 5 clusters

Technique	Parameter settings	$PM_{W.A.}$	PM_{A}	$R_{W.A.}$	$P_{W.A.}$	$F1_{W.A.}$	$F2_{W.A.}$	P/T-CD$_{A}$	P/T-CD$_{W.A.}$	Run time
ActiTraC^{freq}	mcs25tf100	0,884	0,361	0,974	0,802	0,849	0,903	3,966	2,922	00:07:52,207
ActiTraC^{freq}	mcs50tf100	0,888	0,318	0,973	0,532	0,684	0,829	4,036	4,119	00:50:27,167
ActiTraC^{freq}	mcs100tf100	0,888	0,318	0,973	0,532	0,684	0,829	4,036	4,119	00:50:20,917
ActiTraC^{freq}	mcs25tf95	0,770	0,245	0,976	0,804	0,857	0,913	3,909	2,914	00:04:08,884
ActiTraC^{freq}	mcs50tf95	0,167	0,101	0,950	0,344	0,504	0,700	4,552	5,048	00:13:16,933
ActiTraC^{freq}	mcs100tf95	0,177	0,099	0,949	0,321	0,480	0,682	4,119	5,400	01:11:13,640
ActiTraC^{MRA}	w25mcs25tf100	0,850	0,332	0,965	0,730	0,793	0,868	3,813	3,222	00:11:38,654
ActiTraC^{MRA}	w25mcs50tf100	0,831	0,295	0,960	0,672	0,762	0,852	4,062	3,718	00:10:42,608
ActiTraC^{MRA}	w25mcs100tf100	0,848	0,293	0,965	0,511	0,665	0,814	4,121	4,479	00:46:54,615
ActiTraC^{MRA}	w50mcs25tf100	0,816	0,356	0,955	0,742	0,808	0,876	3,280	2,880	00:08:28,628
ActiTraC^{MRA}	w50mcs50tf100	0,818	0,281	0,955	0,561	0,698	0,827	3,798	3,813	00:18:53,143
ActiTraC^{MRA}	w50mcs100tf100	0,831	0,271	0,959	0,538	0,687	0,826	3,824	3,892	00:37:59,973
ActiTraC^{MRA}	w100mcs25tf100	0,816	0,356	0,955	0,744	0,810	0,877	3,270	2,873	00:08:26,081
ActiTraC^{MRA}	w100mcs50tf100	0,818	0,284	0,955	0,563	0,700	0,829	3,791	3,804	00:18:38,550
ActiTraC^{MRA}	w100mcs100tf100	0,831	0,271	0,959	0,538	0,687	0,826	3,824	3,892	00:37:43,442
ActiTraC^{MRA}	w25mcs25tf95	0,070	0,064	0,961	0,391	0,554	0,742	4,605	5,135	00:13:38,573
ActiTraC^{MRA}	w25mcs50tf95	0,065	0,062	0,962	0,375	0,539	0,731	4,455	5,200	00:18:46,487
ActiTraC^{MRA}	w25mcs100tf95	0,029	0,039	0,957	0,336	0,496	0,696	4,586	5,688	00:47:12,912
ActiTraC^{MRA}	w50mcs25tf95	0,627	0,111	0,955	0,483	0,631	0,786	5,449	6,186	00:10:00,625
ActiTraC^{MRA}	w50mcs50tf95	0,642	0,077	0,960	0,511	0,639	0,777	5,347	7,168	00:14:56,117
ActiTraC^{MRA}	w50mcs100tf95	0,048	0,055	0,960	0,340	0,499	0,698	4,983	7,098	00:49:55,830
ActiTraC^{MRA}	w100mcs25tf95	0,577	0,089	0,954	0,472	0,621	0,778	5,174	6,036	00:10:06,485
ActiTraC^{MRA}	w100mcs50tf95	0,653	0,080	0,957	0,517	0,648	0,785	5,203	7,159	00:14:48,759
ActiTraC^{MRA}	w100mcs100tf95	0,101	0,074	0,958	0,343	0,504	0,704	4,372	5,923	00:48:49,253
MR	Nominal-EuclideanDistance	0,432	0,084	0,868	0,606	0,672	0,754	6,016	5,642	00:03:31,702
MR	Nominal-FscoreSimilarity	0,409	0,067	0,862	0,545	0,642	0,745	6,069	5,958	00:02:32,132
MR	Numeric-EuclideanDistance	0,432	0,084	0,868	0,606	0,672	0,754	6,016	5,642	00:03:55,876
MR	Numeric-FscoreSimilarity	0,409	0,067	0,862	0,545	0,642	0,745	6,069	5,958	00:02:23,517
MRA	Nominal-EuclideanDistance	0,174	0,040	0,798	0,391	0,506	0,639	5,893	5,735	00:03:18,056
MRA	Nominal-FscoreSimilarity	0,056	0,062	0,792	0,322	0,456	0,610	6,564	6,672	00:02:20,752
MRA	Numeric-EuclideanDistance	0,174	0,040	0,798	0,391	0,506	0,639	5,893	5,735	00:03:10,888
MRA	Numeric-FscoreSimilarity	0,056	0,062	0,792	0,322	0,456	0,610	6,564	6,672	00:02:12,998
GED	-	0,180	0,026	0,839	0,276	0,413	0,593	5,186	6,199	00:01:46,000
LED	-	0,151	0,028	0,853	0,305	0,434	0,603	5,685	6,270	00:01:45,825
BOA	Nominal-EuclideanDistance	0,018	0,009	0,740	0,373	0,489	0,608	5,347	6,627	00:01:21,928
BOA	Nominal-FscoreSimilarity	0,112	0,021	0,812	0,331	0,456	0,607	5,462	5,961	00:01:25,693
BOA	Numeric-EuclideanDistance	0,018	0,009	0,740	0,373	0,489	0,608	5,347	6,627	00:01:23,303
BOA	Numeric-FscoreSimilarity	0,112	0,021	0,812	0,331	0,456	0,607	5,462	5,961	00:01:26,271
3-gram	Nominal-EuclideanDistance	0,047	0,050	0,826	0,446	0,552	0,673	5,017	5,507	00:03:50,472
3-gram	Nominal-FscoreSimilarity	0,222	0,089	0,812	0,519	0,621	0,716	4,929	5,453	00:02:34,266
3-gram	Numeric-EuclideanDistance	0,047	0,050	0,826	0,446	0,552	0,673	5,017	5,507	00:03:43,756
3-gram	Numeric-FscoreSimilarity	0,222	0,089	0,812	0,519	0,621	0,716	4,929	5,453	00:02:44,179
Markov	-	0,456	0,043	0,890	0,514	0,633	0,754	4,983	4,710	01:26:46,000
Random	-	0,106	0,014	0,811	0,340	0,472	0,623	6,193	6,043	00:00:15,562
No clustering	-	0,003	0,006	0,693	0,237	0,353	0,500	7,252	7,252	NA

Table D.12: Results of different trace clustering techniques with varying parameter settings for event log ICP in case of the creation of 5 clusters.

List of Figures

List of Tables

Bibliography

[1] A. Adriansyah, J. Muñoz-Gama, J. Carmona, B. F. van Dongen, and W. M. P. van der Aalst. **Alignment Based Precision Checking**. BPM Center Report BPM-12-10, 2012.

[2] A. Adriansyah, N. Sidorova, and B. F. van Dongen. **Cost-Based Fitness in Conformance Checking**. In B. Caillaud, J. Carmona, and K. Hiraishi, editors, *ACSD*, pages 57–66. IEEE, 2011.

[3] A. Adriansyah, B. F. van Dongen, and W. M. P. van der Aalst. **Towards Robust Conformance Checking**. In M. zur Muehlen and J. Su, editors, *Business Process Management Workshops*, 66 of *Lecture Notes in Business Information Processing*, pages 122–133. Springer, 2010.

[4] A. Adriansyah, B. F. van Dongen, and W. M. P. van der Aalst. **Conformance Checking Using Cost-Based Fitness Analysis**. In *EDOC*, pages 55–64. IEEE Computer Society, 2011.

[5] R. Agrawal, D. Gunopulos, and F. Leymann. **Mining Process Models from Workflow Logs**. In *Proceedings of the 6th International Conference on Extending Database Technology (EDBT'98)*, pages 469–483, 1998.

[6] H. Akaike. **A new look at the statistical model identification**. *IEEE Transactions on Automatic Control*, 19(6):716 – 723, 1974.

[7] A. K. Alves de Medeiros. *Genetic Process Mining*. PhD thesis, TU Eindhoven, 2006.

[8] A. K. Alves de Medeiros, W. M. P. van der Aalst, and A. J. M. M. Weijters. **Quantifying process equivalence based on observed behavior**. *Data & Knowledge Engineering*, 64(1):55–74, 2008.

[9] A. K. Alves de Medeiros, B. F. van Dongen, W. M. P. van der Aalst, and A. J. M. M. Weijters. **Process Mining: Extending the alpha-algorithm to Mine Short Loops**. BETA working paper series 113, TU Eindhoven, 2004.

[10] A. K. Alves de Medeiros, A. J. M. M. Weijters, and W. M. P. van der Aalst. **Genetic process mining: an experimental evaluation**. *Data Mining and Knowledge Discovery*, 14(2):245–304, 2007.

[11] D. Angluin. **Queries and Concept Learning**. *Machine Learning*, 2(4):319–342, 1987.

[12] K. Anyanwu, A. P. Sheth, J. Cardoso, J. A. Miller, and K. Kochut. **Healthcare Enterprise Process Development and Integration**. *Journal of Research and Practice in Information Technology*, 35(2):83–98, 2003.

[13] H. Blockeel and L. De Raedt. **Top-down induction of first-order logical decision trees**. *Artificial Intelligence*, 101(1-2):285–297, 1998.

[14] R. P. J. C. Bose and W. M. P. van der Aalst. **Abstractions in Process Mining: A Taxonomy of Patterns**. In U. Dayal, J. Eder, J. Koehler, and H. A. Reijers, editors, *BPM*, 5701 of *Lecture Notes in Computer Science*, pages 159–175. Springer, 2009.

[15] R. P. J. C. Bose and W. M. P. van der Aalst. **Context Aware Trace Clustering: Towards Improving Process Mining Results**. In *SDM*, pages 401–412. SIAM, 2009.

[16] R. P. J. C. Bose and W. M. P. van der Aalst. **Trace Clustering Based on Conserved Patterns: Towards Achieving Better Process Models**. In S. Rinderle-Ma, S. W. Sadiq, and F. Leymann, editors, *Business Process Management Workshops*, 43 of *Lecture Notes in Business Information Processing*, pages 170–181. Springer, 2009.

[17] R. P. J. C. Bose and W. M. P. van der Aalst. **Analysis of Patient Treatment Procedures**. In F. Daniel, K. Barkaoui, and S. Dustdar, editors, *Business Process Management Workshops (1)*, 99 of *Lecture Notes in Business Information Processing*, pages 165–166. Springer, 2011.

[18] R. P. J. C. Bose and W. M. P. van der Aalst. **Process diagnostics using trace alignment: Opportunities, issues, and challenges**. *Information Systems*, 37(2):117–141, 2012.

[19] M. Bozkaya, J. Gabriels, and J. M. E. M. van der Werf. **Process Diagnostics: A Method Based on Process Mining**. In A. Kusiak and S. goo Lee, editors, *eKNOW*, pages 22–27. IEEE Computer Society, 2009.

[20] C. Bratosin, N. Sidorova, and W. M. P. van der Aalst. **Discovering Process Models with Genetic Algorithms Using Sampling**. In R. Setchi, I. Jordanov, R. J. Howlett, and L. C. Jain, editors, *KES (1)*, 6276 of *Lecture Notes in Computer Science*, pages 41–50. Springer, 2010.

[21] J. C. A. M. Buijs, B. F. van Dongen, and W. M. P. van der Aalst. **A genetic algorithm for discovering process trees**. In *2012 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, june 2012.

[22] J. C. A. M. Buijs, B. F. van Dongen, and W. M. P. van der Aalst. **On the Role of Fitness, Precision, Generalization and Simplicity in Process Discovery (to appear)**. In *20th International Conference on Cooperative Information Systems (CoopIS 2012)*, lncs, 2012.

[23] A. Burattin and A. Sperduti. **Automatic determination of parameters' values for Heuristics Miner++**. In *IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2010.

[24] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. **Model-Based Clustering and Visualization of Navigation Patterns on a Web Site**. *Data Mining and Knowledge Discovery*, 7(4):399–424, 2003.

[25] T. Calders, C. W. Günther, M. Pechenizkiy, and A. Rozinat. **Using minimum description length for process mining**. In S. Y. Shin and S. Ossowski, editors, *SAC*, pages 1451–1455. ACM, 2009.

[26] J. Cardoso. **Evaluating the Process Control-Flow Complexity Measure**. In *ICWS*, pages 803–804. IEEE Computer Society, 2005.

[27] J. Carmona, J. Cortadella, and M. Kishinevsky. **New Region-Based Algorithms for Deriving Bounded Petri Nets**. *IEEE Trans. Comput.*, 59(3):371–384, 2010.

[28] F. Caron, J. Vanthienen, J. De Weerdt, and B. Baesens. **Advanced Care-Flow Mining and Analysis**. In F. Daniel, K. Barkaoui, and S. Dustdar, editors, *Business Process Management Workshops (1)*, 99 of *Lecture Notes in Business Information Processing*, pages 167–168. Springer, 2011.

[29] R. Cattell. **Scree Test For Number Of Factors**. *Multivariate Behavioral Research*, 1(2):245–276, 1966.

[30] W. Cohen. **Fast Effective Rule Induction**. In A. Prieditis and S. Russell, editors, *Proceedings of the 12th International Conference on Machine Learning*, pages 115–123, Tahoe City, CA, 1995. Morgan Kaufmann Publishers.

[31] D. Cohn, L. Atlas, and R. Ladner. **Improving generalization with active learning**. *Machine Learning*, 15:201–221, 1994.

[32] J. E. Cook, Z. Du, C. Liu, and A. L. Wolf. **Discovering Models of Behavior for Concurrent Workflows**. *Computers in Industry*, 53(3):297–319, 2004.

[33] J. E. Cook and A. L. Wolf. **Discovering Models of Software Processes from Event-Based Data**. *ACM Transactions on Software Engineering and Methodology*, 7(3):215–249, 1998.

[34] J. E. Cook and A. L. Wolf. **Software Process Validation: Quantitatively Measuring the Correspondence of a Process to a Model**. *ACM Trans. Softw. Eng. Methodol.*, 8(2):147–176, 1999.

[35] A. Datta. **Automating the Discovery of AS-IS Business Process Models: Probabilistic and Algorithmic Approaches**. *Information Systems Research*, 9(3):275–301, 1998.

[36] T. H. Davenport. *Process innovation: reengineering work through information technology.* Harvard Business School Press, Boston, MA, USA, 1993.

[37] T. H. Davenport. **Competing on Analytics**. *Harvard Business Review*, 84(1):98–107, 2006.

[38] J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens. **A Critical Evaluation Study of Model-Log Metrics in Process Discovery**. In M. zur Muehlen and J. Su, editors, *Business Process Management Workshops*, 66 of *Lecture Notes in Business Information Processing*, pages 158–169. Springer, 2010.

[39] J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens. **A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs**. *Information Systems*, 37(7):654–676, 2012.

[40] J. De Weerdt, A. Schupp, A. Vanderloock, and B. Baesens. **Process Mining for the multi-faceted analysis of business processes - A case study in a financial services organization**. *Computers in Industry*, Forthcoming.

[41] J. De Weerdt, S. vanden Broucke, J. Vanthienen, and B. Baesens. **Active Trace Clustering for Improved Process Discovery**. *Submitted to IEEE Transactions on Knowledge and Data Engineering*, 2012.

[42] J. De Weerdt, F. Caron, J. Vanthienen, and B. Baesens. **Getting a Grasp on Clinical Pathway Data: An Approach Based on Process Mining**. In *The Third Workshop on Data Mining for Healthcare Management at the 16th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Kuala Lumpur, Malaysia (forthcoming)*, 2012.

[43] J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens. **A robust F-measure for evaluating discovered process models**. In *CIDM*, pages 148–155. IEEE, 2011.

[44] J. De Weerdt, S. vanden Broucke, J. Vanthienen, and B. Baesens. **Leveraging process discovery with trace clustering and text mining for intelligent analysis of incident management processes**. In *IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2012.

[45] R. Dechter and J. Pearl. **Generalized Best-First Search Strategies and the Optimality of A***. *J. ACM*, 32(3):505–536, 1985.

[46] J. Demsar. **Statistical Comparisons of Classifiers over Multiple Data Sets**. *Journal of Machine Learning Research*, 7:1–30, 2006.

[47] R. M. Dijkman, M. Dumas, B. F. van Dongen, R. Käärik, and J. Mendling. **Similarity of business process models: Metrics and evaluation**. *Information Systems*, 36(2):498–516, 2011.

[48] M. Dumas, W. M. P. van der Aalst, and A. H. M. ter Hofstede. *Process-Aware Information Systems: Bridging People and Software through Process Technology*. John Wiley & Sons, Inc., 2005.

[49] O. J. Dunn. **Multiple Comparisons Among Means**. *Journal of the American Statistical Association*, 56(293):52–64, 1961.

[50] J. G. Dy and C. E. Brodley. **Feature Selection for Unsupervised Learning**. *J. Mach. Learn. Res.*, 5:845–889, 2004.

[51] T. Fawcett. **An introduction to ROC analysis**. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[52] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. **Knowledge Discovery and Data Mining: Towards a Unifying Framework**. In *KDD*, pages 82–88, 1996.

[53] D. R. Ferreira and D. Gillblad. **Discovering Process Models from Unlabelled Event Logs**. In U. Dayal, J. Eder, J. Koehler, and H. A. Reijers, editors, *BPM*, 5701 of *Lecture Notes in Computer Science*, pages 143–158. Springer, 2009.

[54] D. R. Ferreira, M. Zacarias, M. Malheiros, and P. Ferreira. **Approaching Process Mining with Sequence Clustering: Experiments and Findings**. In G. Alonso, P. Dadam, and M. Rosemann, editors, *BPM*, 4714 of *Lecture Notes in Computer Science*, pages 360–374. Springer, 2007.

[55] D. R. Ferreira, M. Zacarias, M. Malheiros, and P. Ferreira. **Approaching Process Mining with Sequence Clustering: Experiments and Findings**. In G. Alonso, P. Dadam, and M. Rosemann, editors, *BPM*, 4714 of *Lecture Notes in Computer Science*, pages 360–374. Springer, 2007.

[56] H. Ferreira and D. R. Ferreira. **An integrated life cycle for workflow management based on learning and planning**. *International Journal of Cooperative Information Systems*, 15(4):485–505, 2006.

[57] F. Folino, G. Greco, A. Guzzo, and L. Pontieri. **Discovering expressive process models from noised log data**. In B. C. Desai, D. Saccà, and S. Greco, editors, *IDEAS*, ACM International Conference Proceeding Series, pages 162–172. ACM, 2009.

[58] F. Folino, G. Greco, A. Guzzo, and L. Pontieri. **Mining usage scenarios in business processes: Outlier-aware discovery and runtime prediction**. *Data Knowl. Eng.*, 70(12):1005–1029, 2011.

[59] M. Friedman. **A Comparison of Alternative Tests of Significance for the Problem of m Rankings**. *The Annals of Mathematical Statistics*, 11(1):86–92, 1940.

[60] W. Gaaloul, K. Baïna, and C. Godart. **Towards Mining Structural Workflow Patterns**. In K. V. Andersen, J. K. Debenham, and R. Wagner, editors, *DEXA*, 3588 of *Lecture Notes in Computer Science*, pages 24–33. Springer, 2005.

[61] S. Goedertier. *Declarative Techniques for Modeling and Mining Business Processes*. Phd thesis, Katholieke Universiteit Leuven, Faculty of Business and Economics, Leuven, September 2008.

[62] S. Goedertier, J. De Weerdt, D. Martens, J. Vanthienen, and B. Baesens. **Process discovery in event logs: An application in the telecom industry**. *Appl. Soft Comput.*, 11(2):1697–1710, 2011.

[63] S. Goedertier, D. Martens, J. Vanthienen, and B. Baesens. **Robust Process Discovery with Artificial Negative Events**. *Journal of Machine Learning Research*, 10:1305–1340, 2009.

[64] M. Golfarelli, S. Rizzi, and I. Cella. **Beyond data warehousing: what's next in business intelligence?** In I.-Y. Song and K. C. Davis, editors, *DOLAP*, pages 1–6. ACM, 2004.

[65] G. Greco, A. Guzzo, and L. Pontieri. **Mining taxonomies of process models**. *Data Knowl. Eng.*, 67(1):74–102, 2008.

[66] G. Greco, A. Guzzo, L. Pontieri, and D. Saccà. **Discovering Expressive Process Models by Clustering Log Traces**. *IEEE Trans. Knowl. Data Eng.*, 18(8):1010–1027, 2006.

[67] C. W. Günther and W. M. P. van der Aalst. **Fuzzy Mining - Adaptive Process Simplification Based on Multi-perspective Metrics**. In G. Alonso, P. Dadam, and M. Rosemann, editors, *BPM*, 4714 of *Lecture Notes in Computer Science*, pages 328–343. Springer, 2007.

[68] C. W. Günther. *Process Mining in Flexible Environments*. PhD thesis, TU Eindhoven, 2009.

[69] C. W. Günther. **XES Standard Definition**. www.xes-standard.org, 2009.

[70] M. Hammer and J. Champy. *Reengineering the corporation: a manifesto for business revolution*. HarperBusiness, New York, NY, 1993.

[71] J. Herbst and D. Karagiannis. **Workflow mining with InWoLvE**. *Computers in Industry*, 53(3):245–264, 2004.

[72] T. Hofmann and J. M. Buhmann. **Active Data Clustering**. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *NIPS*. The MIT Press, 1997.

[73] H. Hotelling. **Relations Between Two Sets of Variates**. *Biometrika*, 28(3/4):321–377, 1936.

[74] M. E. Houle, H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek. **Can Shared-Neighbor Distances Defeat the Curse of Dimensionality?** In M. Gertz and B. Ludäscher, editors, *SSDBM*, 6187 of *Lecture Notes in Computer Science*, pages 482–500. Springer, 2010.

[75] Y. Hu. **Algorithms for Visualizing Large Networks**. in Combinatorial Scientific Computing, eds. U. Naumann and O. Schenk, to appear.

[76] M. Jans, J. M. van der Werf, N. Lybaert, and K. Vanhoof. **A business process mining application for internal transaction fraud mitigation**. *Expert Systems with Applications*, 38(10):13351 – 13359, 2011.

[77] I. T. Jolliffe. *Principal Component Analysis*. Springer, second edition, 2002.

[78] L. T. Kohn, J. M. Corrigan, and M. S. Donaldson. *To Err Is Human: Building a Safer Health System*. The National Academies Press, Washington DC, 2000. Committee on Quality of Health Care in America, Institute of Medicine.

[79] H.-P. Kriegel, K. M. Borgwardt, P. Kröger, A. Pryakhin, M. Schubert, and A. Zimek. **Future trends in data mining**. *Data Min. Knowl. Discov.*, 15(1):87–97, 2007.

[80] J. Lafferty, A. McCallum, and F. Pereira. **Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data**. In C. E. Brodley and A. P. Danyluk, editors, *Proceedings of the 18th International Conference on Machine Learning (ICML 2001)*, pages 282–289. Morgan Kaufmann, 2001.

[81] E. Lamma, P. Mello, M. Montali, F. Riguzzi, and S. Storari. **Inducing Declarative Logic-Based Models from Labeled Traces**. In G. Alonso, P. Dadam, and M. Rosemann, editors, *BPM*, 4714 of *Lecture Notes in Computer Science*, pages 344–359. Springer, 2007.

[82] K. B. Lassen and W. M. P. van der Aalst. **Complexity metrics for Workflow nets**. *Information & Software Technology*, 51(3):610–626, 2009.

[83] R. Lenz, T. Elstner, H. Siegele, and K. A. Kuhn. **A Practical Approach to Process Support in Health Information Systems**. *Journal of the American Medical Informatics Association*, 9(6):571–585, 2002.

[84] R. Lenz and M. Reichert. **IT support for healthcare processes - premises, challenges, perspectives**. *Data Knowl. Eng.*, 61(1):39–58, 2007.

[85] V. I. Levenshtein. **Binary Codes Capable of Correcting Deletions, Insertions and Reversals**. *Soviet Physics Doklady*, 10:707–710, 1966.

[86] F. Liu. *Fault Diagnosis Using Process Mining.* Master's thesis, Technische Universiteit Eindhoven, 2010.

[87] F. M. Maggi, A. J. Mooij, and W. M. P. van der Aalst. **User-guided discovery of declarative process models**. In *CIDM*, pages 192–199. IEEE, 2011.

[88] H. Mannila and C. Meek. **Global partial orders from sequential data**. In *Proceedings of the 6th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '00)*, pages 161–168, New York, NY, USA, 2000. ACM.

[89] R. S. Mans, H. Schonenberg, M. Song, W. M. P. van der Aalst, and P. J. M. Bakker. **Application of Process Mining in Healthcare - A Case Study in a Dutch Hospital**. In A. L. N. Fred, J. Filipe, and H. Gamboa, editors, *BIOSTEC (Selected Papers)*, 25 of *Communications in Computer and Information Science*, pages 425–438. Springer, 2008.

[90] D. Martens, B. Baesens, and T. V. Gestel. **Decompositional Rule Extraction from Support Vector Machines by Active Learning**. *IEEE Trans. Knowl. Data Eng.*, 21(2):178–191, 2009.

[91] L. Maruster, A. J. M. M. Weijters, W. M. P. van der Aalst, and A. van den Bosch. **A Rule-Based Approach for Process Discovery: Dealing with Noise and Imbalance in Process Logs.** *Data Mining and Knowledge Discovery*, 13(1):67–87, 2006.

[92] T. J. McCabe. **A Complexity Measure**. *IEEE Trans. Software Eng.*, 2(4):308–320, 1976.

[93] J. Mendling, H. A. Reijers, and J. Cardoso. **What Makes Process Models Understandable?** In G. Alonso, P. Dadam, and M. Rosemann, editors, *BPM*, 4714 of *Lecture Notes in Computer Science*, pages 48–63. Springer, 2007.

[94] J. Mendling, H. A. Reijers, and W. M. P. van der Aalst. **Seven process modeling guidelines (7PMG)**. *Information & Software Technology*, 52(2):127–136, 2010.

[95] S. Muggleton. **Inductive Logic Programming**. *New Generation Comput.*, 8(4):295–318, 1991.

[96] J. Muñoz-Gama and J. Carmona. **A Fresh Look at Precision in Process Conformance**. In R. Hull, J. Mendling, and S. Tai, editors, *Business Process Management*, 6336 of *Lecture Notes in Computer Science*, pages 211–226. Springer, 2010.

[97] J. Muñoz-Gama and J. Carmona. **Enhancing precision in Process Conformance: Stability, confidence and severity**. In *CIDM*, pages 184–191. IEEE, 2011.

[98] T. Murata. **Petri Nets: Properties, Analysis and Applications**. *Proceedings of the IEEE*, 77(4):541–580, 1989.

[99] National Library of Medicine. **Medical Subject Heading (MeSH)**. `http://www.ncbi.nlm.nih.gov/mesh`, 1996.

[100] Object Management Group (OMG). **Business Process Modeling Notation (BPMN) – Spectification**. OMG Document – formal/2011-01-03, January 2011.

[101] OECD. **OECD Health Data 2010: Statistics and Indicators**. `http://www.oecd.org/health/healthdata`, 2010.

[102] M. Pesic and W. M. P. van der Aalst. **A Declarative Approach for Flexible Business Processes Management**. In J. Eder and S. Dustdar, editors, *Business Process Management Workshops*, 4103 of *Lecture Notes in Computer Science*, pages 169–180. Springer, 2006.

[103] J. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[104] L. R. Rabiner. **A tutorial on hidden markov models and selected applications in speech recognition**. In *Proceedings of the IEEE*, pages 257–286, 1989.

[105] Á. Rebuge and D. R. Ferreira. **Business process analysis in healthcare environments: A methodology based on process mining**. *Information Systems*, 37(2):99–116, 2012.

[106] H. A. Reijers, N. Russell, S. van der Geer, and G. A. M. Krekels. **Workflow for Healthcare: A Methodology for Realizing Flexible Medical Treatment Processes**. In S. Rinderle-Ma, S. W. Sadiq, and F. Leymann, editors, *Business Process Management Workshops*, 43 of *Lecture Notes in Business Information Processing*, pages 593–604. Springer, 2009.

[107] A. Rozinat and W. M. P. van der Aalst. **Conformance checking of processes based on monitoring real behavior**. *Information Systems*, 33(1):64–95, 2008.

[108] A. Rozinat, M. Veloso, and W. M. P. van der Aalst. **Using Hidden Markov Models to Evaluate the Quality of Discovered Process Models.** BPM Center Report BPM-08-10, 2008.

[109] A. Rozinat. *Process Mining: Conformance and Extension.* PhD thesis, TU Eindhoven, 2010.

[110] A. Rozinat, I. S. M. de Jong, C. W. Günther, and W. M. P. van der Aalst. **Process Mining Applied to the Test Process of Wafer Scanners in ASML**. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 39(4):474–479, 2009.

[111] A. Rozinat, A. K. A. de Medeiros, C. W. Günther, A. J. M. M. Weijters, and W. M. P. van der Aalst. **The Need for a Process Mining Evaluation Framework in Research and Practice**. In G. Alonso, P. Dadam, and M. Rosemann, editors, *Business Process Management Workshops*, 4714 of *Lecture Notes in Computer Science*, pages 84–89. Springer, 2007.

[112] A. Rozinat, R. S. Mans, M. Song, and W. M. P. van der Aalst. **Discovering simulation models**. *Information Systems*, 34(3):305–327, 2009.

[113] A. Rozinat and W. M. P. van der Aalst. **Decision Mining in ProM**. In S. Dustdar, J. L. Fiadeiro, and A. P. Sheth, editors, *Business Process Management*, 4102 of *Lecture Notes in Computer Science*, pages 420–425. Springer, 2006.

[114] G. Schimm. **Process Miner - A Tool for Mining Process Schemes from Event-Based Data**. In S. Flesca, S. Greco, N. Leone, and G. Ianni, editors, *JELIA*, 2424 of *Lecture Notes in Computer Science*, pages 525–528. Springer, 2002.

[115] G. Schimm. **Mining exact models of concurrent workflows**. *Computers in Industry*, 53(3):265 – 281, 2004.

[116] G. E. Schwarz. **Estimating the Dimension of a Model**. *The Annals of Statistics*, 6(2):pp. 461–464, 1978.

[117] B. Settles. **Active Learning Literature Survey**. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[118] M. Sole and J. Carmona. **Region-Based Foldings in Process Discovery**. *IEEE Trans. Knowl. Data Eng.*, 99(PrePrints), 2011.

[119] M. Solé and J. Carmona. **An SMT-Based Discovery Algorithm for C-Nets**. In S. Haddad and L. Pomello, editors, *Petri Nets*, 7347 of *Lecture Notes in Computer Science*, pages 51–71. Springer, 2012.

[120] M. Song, C. W. Günther, and W. M. P. van der Aalst. **Trace Clustering in Process Mining**. In D. Ardagna, M. Mecella, and J. Yang, editors, *Business Process Management Workshops*, 17 of *Lecture Notes in Business Information Processing*, pages 109–120. Springer, 2008.

[121] M. Song and W. M. P. van der Aalst. **Towards comprehensive support for organizational mining**. *Decision Support Systems*, 46(1):300–317, 2008.

[122] D. Stewart and W. Love. **A General Canonical Correlation Index**. *Psychological Bulletin*, 70(3):160–163, 1968. Part 1.

[123] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

[124] W. M. P. van der Aalst. **The Application of Petri Nets to Workflow Management.** *Journal of Circuits, Systems, and Computers*, 8(1):21–66, 1998.

[125] W. M. P. van der Aalst. **Challenges in Business Process Mining.** BPM Center Report BPM-10-01, 2010.

[126] W. M. P. van der Aalst, H. A. Reijers, and M. Song. **Discovering Social Networks from Event Logs**. *Computer Supported Cooperative Work*, 14(6):549–593, 2005.

[127] W. M. P. van der Aalst, H. A. Reijers, A. J. M. M. Weijters, B. F. van Dongen, A. K. Alves de Medeiros, M. Song, and H. M. W. Verbeek. **Business process mining: An industrial application**. *Information Systems*, 32(5):713–732, 2007.

[128] W. M. P. van der Aalst, V. Rubin, B. F. van Dongen, E. Kindler, , and C. W. Günther. **Process Mining: A Two-Step Approach using Transition Systems and Regions**. BPM-06-30, BPM Center Report, 2006.

[129] W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske. **Business Process Management: A Survey.** In W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske, editors, *Business Process Management*, 2678 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2003.

[130] W. M. P. van der Aalst and A. J. M. M. Weijters. **Process mining: a research agenda**. *Computers in Industry*, 53(3):231–244, 2004.

[131] W. M. P. van der Aalst, A. J. M. M. Weijters, and L. Maruster. **Workflow Mining: Discovering Process Models from Event Logs**. *IEEE Trans. Knowl. Data Eng.*, 16(9):1128–1142, 2004.

[132] W. M. P. van der Aalst. **Business Process Management Demystified: A Tutorial on Models, Systems and Standards for Workflow Management**. In J. Desel, W. Reisig, and G. Rozenberg, editors, *Lectures on Concurrency and Petri Nets*, 3098 of *Lecture Notes in Computer Science*, pages 21–58. Springer Berlin / Heidelberg, 2004.

[133] W. M. P. van der Aalst. **Do Petri Nets Provide the Right Representational Bias for Process Mining?** In J. Desel and E. A. Yakovlev, editors, *Workshop Applications of Region Theory 2011 (ART 2011)*, pages 85–94, 2011.

[134] W. M. P. van der Aalst. **On the Representational Bias in Process Mining**. In S. Reddy and S. Tata, editors, *WETICE*, pages 2–7. IEEE Computer Society, 2011.

[135] W. M. P. van der Aalst. *Process Mining - Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.

[136] W. M. P. van der Aalst, A. Adriansyah, and B. F. van Dongen. **Replaying history on process models for conformance checking and performance analysis**. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(2):182–192, 2012.

[137] W. M. P. van der Aalst, H. T. de Beer, and B. F. van Dongen. **Process Mining and Verification of Properties: An Approach Based on Temporal Logic**. In R. Meersman, Z. Tari, M.-S. Hacid, J. Mylopoulos, B. Pernici, Ö. Babaoglu, H.-A. Jacobsen, J. P. Loyall, M. Kifer, and S. Spaccapietra, editors, *OTM Conferences (1)*, 3760 of *Lecture Notes in Computer Science*, pages 130–147. Springer, 2005.

[138] W. M. P. van der Aalst, M. Pesic, and H. Schonenberg. **Declarative workflows: Balancing between flexibility and support**. *Computer Science - R&D*, 23(2):99–113, 2009.

[139] W. M. P. van der Aalst, V. Rubin, H. M. W. Verbeek, B. F. van Dongen, E. Kindler, and C. W. Günther. **Process mining: a two-step approach to balance between underfitting and overfitting**. *Software and System Modeling*, 9(1):87–111, 2010.

[140] W. M. P. van der Aalst, M. H. Schonenberg, and M. Song. **Time prediction based on process mining**. *Information Systems*, 36(2):450–475, 2011.

[141] W. M. P. van der Aalst, B. F. van Dongen, C. W. Günther, A. Rozinat, E. Verbeek, and T. Weijters. **ProM: The Process Mining Toolkit**. In A. K. A. de Medeiros and B. Weber, editors, *BPM (Demos)*, 489 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.

[142] W. M. P. van der Aalst, K. M. van Hee, J. M. E. M. van der Werf, A. Kumar, and M. Verdonk. **Conceptual model for online auditing**. *Decision Support Systems*, 50(3):636–647, 2011.

[143] W. M. P. van der Aalst, K. M. van Hee, J. M. E. M. van der Werf, and M. Verdonk. **Auditing 2.0: Using Process Mining to Support Tomorrow's Auditor**. *IEEE Computer*, 43(3):90–93, 2010.

[144] W. M. P. van der Aalst, M. Weske, and D. Grünbauer. **Case handling: a new paradigm for business process support**. *Data Knowl. Eng.*, 53(2):129–162, 2005.

[145] J. M. E. M. van der Werf, B. F. van Dongen, C. A. J. Hurkens, and A. Serebrenik. **Process Discovery using Integer Linear Programming**. *Fundam. Inform.*, 94(3-4):387–412, 2009.

[146] B. F. van Dongen and W. M. P. van der Aalst. **Multi-Phase Process Mining: Aggregating Instance Graphs into EPCs and Petri Nets**. In *Proceedings of the 2nd International Workshop on Applications of Petri Nets to Coordination, Workflow and Business Process Management (PNCWB)*, 2005.

[147] C. J. van Rijsbergen. *Information Retrieval*. Butterworth, 1979.

[148] S. vanden Broucke, J. De Weerdt, B. Baesens, and J. Vanthienen. **Improved Artificial Negative Event Generation to Enhance Process Event Logs**. In J. Ralyté, X. Franch, S. Brinkkemper, and S. Wrycza, editors, *CAiSE*, 7328 of *Lecture Notes in Computer Science*, pages 254–269. Springer, 2012.

[149] G. M. Veiga and D. R. Ferreira. **Understanding Spaghetti Models with Sequence Clustering for ProM**. In S. Rinderle-Ma, S. W. Sadiq, and F. Leymann, editors, *Business Process Management Workshops*, 43 of *Lecture Notes in Business Information Processing*, pages 92–103. Springer, 2009.

[150] G. M. Veiga and D. R. Ferreira. **Understanding Spaghetti Models with Sequence Clustering for ProM**. In S. Rinderle-Ma, S. W. Sadiq, and F. Leymann, editors, *Business Process Management Workshops*, 43 of *Lecture Notes in Business Information Processing*, pages 92–103. Springer, 2010.

[151] M. Weidlich, A. Polyvyanyy, N. Desai, J. Mendling, and M. Weske. **Process compliance analysis based on behavioural profiles**. *Inf. Syst.*, 36(7):1009–1025, 2011.

[152] A. J. M. M. Weijters and W. M. P. van der Aalst. **Rediscovering workflow models from event-based data using little thumb**. *Integrated Computer-Aided Engineering*, 10(2):151–162, 2003.

[153] A. J. M. M. Weijters, W. M. P. van der Aalst, and A. K. Alves de Medeiros. **Process Mining with the HeuristicsMiner algorithm**. BETA working paper series 166, TU Eindhoven, 2006.

[154] L. Wen, W. M. P. van der Aalst, J. Wang, and J. Sun. **Mining process models with non-free-choice constructs**. *Data Mining and Knowledge Discovery*, 15(2):145–180, 2007.

[155] L. Wen, J. Wang, W. M. P. van der Aalst, B. Huang, and J. Sun. **A novel approach for process mining based on event types**. *J. Intell. Inf. Syst.*, 32(2):163–190, 2009.

[156] H. Yang, A. H. M. ter Hofstede, B. F. van Dongen, M. T. Wynn, and J. Wang. **On Global Completeness of Event Logs.** BPM Center Report BPM-10-09, 2010.

[157] H. Yang, B. F. van Dongen, A. H. M. ter Hofstede, M. T. Wynn, and J. Wang. **Estimating Completeness of Event Logs.** BPM Center Report BPM-12-04, 2012.

Doctoral dissertations from the faculty of business and economics

A full list of the doctoral dissertations from the Faculty of Business and Economics can be found at:

www.kuleuven.be/doctoraatsverdediging/archief.htm.