h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

fbmn
FACHBEREICH MATHEMATIK
UND NATURWISSENSCHAFTEN

Hochschule Darmstadt
- Fachbereich Mathematik -

Data-driven process discovery

Reproduktion und kritische Betrachtung der Ergebnisse von F. Mannhardt et al.(TU/e)

Abschlussarbeit im Hauptseminar - Hot Topics in Data Science
vorgelegt von

Lisa Stolz

Referent:	Prof. Dr. Markus Döhring
Korreferentin:	Prof. Dr. Markus Döhring
Ausgabedatum:	June 10, 2018
Abgabedatum:	June 10, 2018

Declaration of Authorship

Ich, Lisa Stolz, versichere hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen. Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, den June 10, 2018

(Vollständige, handschriftliche Unterschrift)

HOCHSCHULE DARMSTADT

Abstract

Faculty Name
Fachbereich Mathematik

Master of Science (M.Sc.)

Data-driven process discovery

by Lisa Stolz

Ziel dieser Seminararbeit ist die kritische Auseinandersetzung mit dem Paper "Data-driven process discovery: revealing conditional infrequent behavior from event logs“ von F. Mannhardt et. al. der Technischen Univerität Eindhoven.
Die Autoren stellen in dieser Arbeit den „Data-aware Heuristic Miner“ (DHM) vor, welcher bisherige heuristische Process Mining Ansätze um ein Maß der bedingten Abhängigkeit erweitert.
Der DHM basiert auf der Idee, dass seltenes - jedoch durch Daten gestütztes - Prozessverhalten nicht als Hintergrundrauschen verworfen werden sollte. Die Autoren erklären in ihrer Arbeit zunächst die theoretischen Hintergründe des DHM und zeigen anschließend die Ergebnisse der Evaluierung in ProM.

In dieser Arbeit wird zunächst ein kurzer Überblick über den DHM und die Erweiterung bisheriger heuristischer Ansätze des Process Mining gegeben.[1] Der Schwerpunkt liegt anschließend auf der praktischen Anwendung des Pakets „DataAwareC-NetMiner“ in ProM, einem interaktiven Tool, das einen schnellen Vergleich mit dem „standard Flexible Heuristic Miner“ erlaubt.
Zunächst werden die im Paper dargestellten Evaluierungen reproduziert und getestet ob die Ergebnisse nachvollzogen werden können.
Die erste Evaluierung basiert auf einem synthetisch generierten Datensatz und die zweite auf realen Event Logs zu Strafzetteln im italienischen Straßenverkehr und Krankenhausrechnungen eines Enterprise Resource Planing Tools.
Wenn die Ergebnisse und das Vorgehen der Autoren reproduziert werden können, werden im Anschluss die Versuchsparameter variiert (z.B. Klassifizierung durch Entscheidungsbäume C3.4 und Objektivitätsmaß Cohens Kappa). Schließlich werden die bis dahin getesteten Mining Methoden auf einen neuen Eventlog Datensatz angewendet und mit den bisherigen Ergebnissen verglichen.

[1] Man17.

Contents

List of Figures

List of Tables

List of Abbreviations

DHM	**D**ata-aware **H**euristic **M**iner
HM	**H**euristic **M**iner
IM	**I**nductive **M**iner

List of Symbols

$C = (\Sigma, s_i, s_o, D, I, O)$	Causal nets (C-nets tuple)
Σ	finite set of activities
$s_i \in \Sigma$	unique start activity
$s_o \in \Sigma$	unique end activity
$D \subseteq \Sigma \times \Sigma$	dependency relation
$B = \{X \subseteq \mathscr{P}(\Sigma) \mid X = \{\varnothing\} \vee \varnothing \notin X\}$	possible bindings
$I \in \Sigma \to B$	set of input bindings per activity
$O \in \Sigma \to B$	set of output bindings per activity
A	attributes
U	values
$L = (E, \Sigma, \#, L)$	event log
E	finite set of unique event identifiers
$\Sigma \subseteq U$	a finite set of activities
$\# : E \to (A \nrightarrow U)$	obtains the attribute values recorded for an event
$\mathscr{L} \subseteq E^*$	the set of traces over E
$\sigma \in L$	one trace - sequence of events for one process instance

Chapter 1

Introduction

Process Models are everywhere - not only in the business world, but they can also be found in social networks, politics and official bodies or Technology. There are two main approaches to a process model. Either they are written in advance to be followed i.e. by employees in a company that introduces new processes to comply with regulatory standard, or they are derived from existing behavior. The latter one describes the field of Process Mining, which relies on digital traces of actions.

The general idea to inspect the data and discover that certain actions follow each other is straight forward, however it becomes more difficult once there are several steps which can follow one action. Are these actions executed in parallel or only one of them exclusively? Do they have to be completed in a certain order?
Even more complex is the question if rare events and actions which can be found in the data are part of the process or just random noise in the sense that they don't belong in the model and can be discarded.

In their paper "Data-driven process discovery: revealing conditional infrequent behavior from event logs", F. Mannhardt et. al. from the Eindhoven University of Technology describe their approach to handle rare events in the data. They introduce the „Data-aware Heuristic Miner" (DHM) which applies classification techniques in order to derive dependencies between activities and thus distinguish between noise and infrequent behavior.

This seminar paper gives an insight to the approach of F. Mannhardt et. al and critically evaluates it.
In the first section the basic concepts of Business Process Mining will be explained - based on concepts applied by the authors. After explaining the theoretical concept of the DHM the results of the empirical evaluation from the paper shall be reproduced and presented in section two.
For evaluation the free interactive tool „ProM" was used and for the replication the package „DataAwareCNetMiner" will be applied. As this package underwent further development after the publication of the described paper, there are is a discrepancy in the results. In Section three the evolved Inductive Data-aware Heuristic Miner will be evaluated on basis of a second paper and compared to the basic approach of the DHM.

Chapter 2

Business Process Mining

2.1 Basic Concepts of Process Mining

Before introducing the Data-aware Heuristic Miner, the basic concepts of Process Mining will be introduced in order to provide a good understanding of Business Process Mining beforehand.
The work in the field of Processes can be seen as a process it self. In order to distinguish the focus of this seminar paper from other fields of research, the main steps are displayed in the following figure.

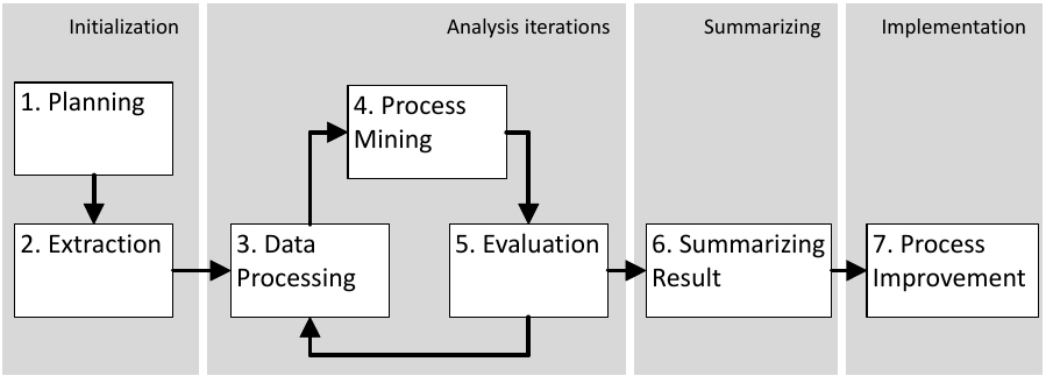


FIGURE 2.1: Steps in Process Mining[Bui17]

The DHM is a new mining method and thus the main focus lies on step 4. and only partly on 3. and 5. when reproducing the results and testing further functionalities in ProM. [Bui17]

2.1.1 Event logs, Petri nets and model criteria

Event logs store information about **activities** and each execution of a process produces a sequence of **events**. Events are stored with attributes A and values U in traces $\sigma \in \mathscr{L}$.
Thus an event log $L = (E, \Sigma, \#, L)$ consists of:

- E - a finite set of unique event identifiers
- $\Sigma \subseteq U$ - a finite set of activities
- $\# : E \to (A \nrightarrow U)$ - attribute values recorded for an event
- $\mathscr{L} \subseteq E^*$ - the set of traces over E

- $\sigma \in \mathscr{L}$ - traces, which record the sequence of events for one process instance - each event occurs only in a single trace

In the following figure this can be seen for three traces with attributes **act**ivity, **p**riority, **n**urse and **t**ype. The Hospital Billing Example will be used through this paper for a better explanation of Data Mining as well as an evaluation data set in chapter 3. [Man17]

(a) Trace $\sigma_1 \in \mathscr{L}$

id	act	p	n	t
e_{11}	Triage	Red		
e_{12}	Register		Joe	
e_{13}	Check			
e_{14}	Check			
e_{15}	Check			
e_{16}	Visit			
e_{17}	X-Ray			
e_{18}	F. Visit			ICU
e_{19}	Prepare			

(b) Trace $\sigma_2 \in \mathscr{L}$

id	act	p	n	t
e_{21}	Triage	Red		
e_{22}	Register		Alice	
e_{23}	Check			
e_{24}	X-Ray			
e_{25}	Visit			
e_{26}	Check			
e_{27}	F. Visit			out
e_{28}	Prepare			
e_{29}	Org. Amb.			

(c) Trace $\sigma_3 \in \mathscr{L}$

id	act	p	n	t
e_{31}	Triage	Red		
e_{32}	Register		Joe	
e_{33}	Check			
e_{34}	Visit			
e_{35}	X-Ray			
e_{36}	Check			
e_{37}	Check			
e_{38}	F. Visit			NC
e_{39}	Prepare			

FIGURE 2.2: Three traces of an example process of the emergency ward data [Man17]

While the data is presented in event logs - consisting of traces, the model derived from it, can be displayed in graphical form (i.e. Petri net, C-net, BPMN). Common patterns, which can be displayed by a Petri net are Sequences, Choice [e, f], Parallelism [b, c, d], and Loops.

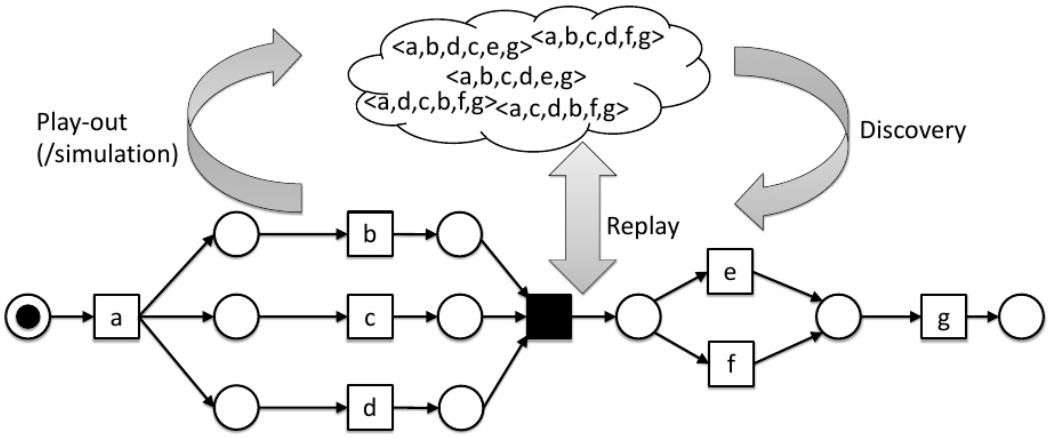


FIGURE 2.3: A Petri net derived from Event logs[Bui17]

Once a model has been discovered the following criteria should be considered:

1. Soundness: Are the criteria for Soundness met?
2. Replay Fitness: Can all traces be represented by the model?
3. Precision: Can the model represent additional cases, not seen in the traces?
4. Generalization: Is the model restrictive or can it be applied in general?

5. Simplicity: Is the model as simple as possible?

Soundness

1. Option to complete
   For each possible state of the process model, it is possible to reach the end state
2. Proper completion
   When the process model reaches the end state, there are no tokens left behind
3. No dead transitions
   Each transition in the process model can be enabled
   [Bui17]

2.1.2 Causal Nets

Conventional model notations like Petri nets and BPMN are often not able to represent observed behavior properly and discovered process models tend to have dead- or livelocks. For this reason C-nets are preferred by the authors as representation for their DHM.
In a Causal-net (C-net) nodes represent activities and arcs represent causal dependencies. Each activity has a set of possible input bindings and a set of possible output bindings. For example in the following Figure the Activity "Register" has one input binding from "Triage" but two sets of output bindings {"Check", "Visit"}, {"Check", "X-Ray"}. This means that "Register" is either followed by "Check" and "Visit" or "Check" and "X-Ray".[Man17][Van+11]

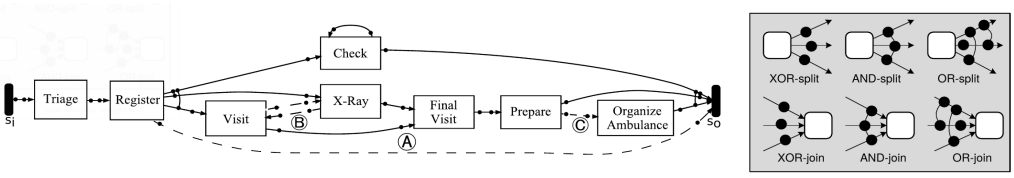


FIGURE 2.4: Splits and Joins in a C-net of the emergency ward example process[Man17][Van+11]

In mathematical notation a C-net is a tuple $C = (\Sigma, s_i, s_o, D, I, O)$, consisting of:

- Σ finite set of activities
- $s_i \in \Sigma$ unique start activity
- $s_o \in \Sigma$ unique end activity
- $D \subseteq \Sigma \times \Sigma$ dependency relation
- $B = \{X \subseteq \mathscr{P}(\Sigma) \mid X = \{\varnothing\} \vee \varnothing \notin X\}$ possible bindings
- $I \in \Sigma \to B$ set of input bindings per activity
- $O \in \Sigma \to B$ set of output bindings per activity
  [Man17]

2.2 Mining Methods

In order to discover an adequate process model the DHM builds on methods, which are applied in several established miners. The concepts of relation notation, footprint- / dependency matrices and classification by decision trees in an event log, will be briefly introduced in connection to the respective miner.

2.2.1 Alpha Miner

This very basic miner was the first to bridge from Event Logs to Petri nets. First a footprint matrix is detected from the event log. By applying the following notation, it can be interpreted from the symmetric matrix, that b follows a, but e, f and g never follow a.

$>$	Directly follows	a $>$ b	a is directly followed by b
$\rightarrow$	Sequence	a $\rightarrow$ b	if $a > b$ and not $b > a$
||	Parallel	a || b	if both a > b and b > a
#	No direct relation	a # b	if neither $a > b$ and $b > a$

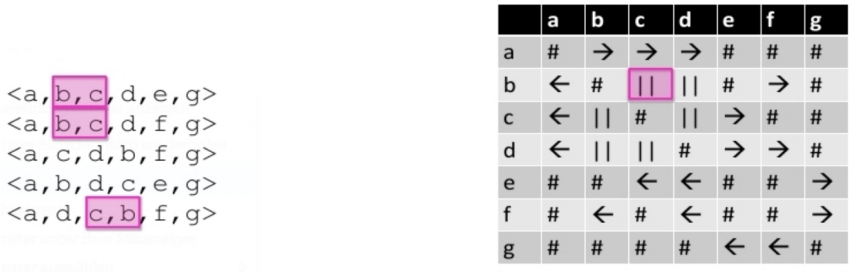


FIGURE 2.5: The Footprint Matrix of the Alpha Miner[Bui17]

From the footprint matrix the following Petri net can be derived. The actions b, c and d, which were marked as parallel in the above Figure, are now displayed as parallel paths.[Bui17]

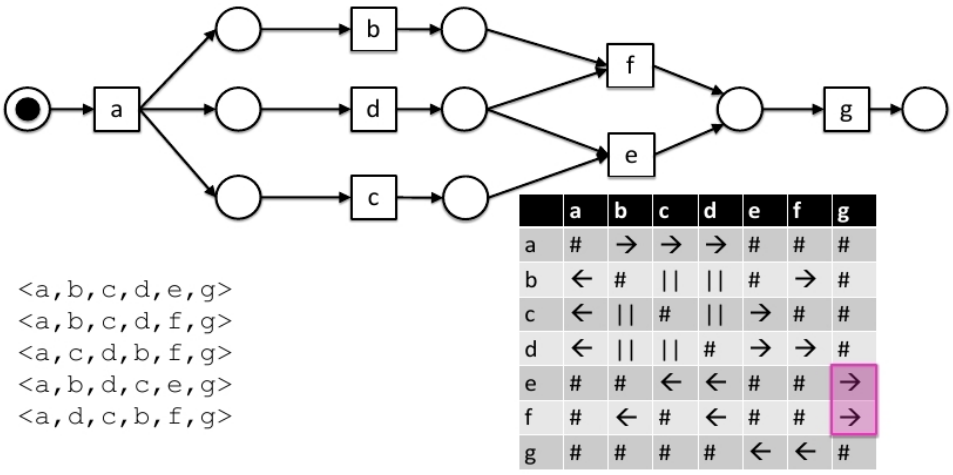


FIGURE 2.6: A derived Petri net of the Footprint Matrix[Bui17]

2.2.2 Heuristics Miner (HM)

The Heuristic Miner is an Improvement of the Alpha miner, as it takes frequencies into account, detects short-loops and can detect skipping activities. However it does not guarantee for sound process models.

The basic idea is, to count the relations in the footprint matrix and calculate relative frequencies which returns a dependency matrix, as depicted in the following Figure.

|=>|	a	b	c	d	e	f	g
a		.98	.67	.80			
b	-.98		.82	.67		.86	
c	-.67	-.82		.90	.92		
d	-.80	-.67	-.90		.95	.97	
e			-.92	-.95			.95
f		-.86		-.97			.98
g					-.95	-.98	

|>|	a	b	c	d	e	f	g
a		56	2	4			
b			44	12		6	
c		4		46	12		
d		2	4		18	38	
e							18
f							44
g							

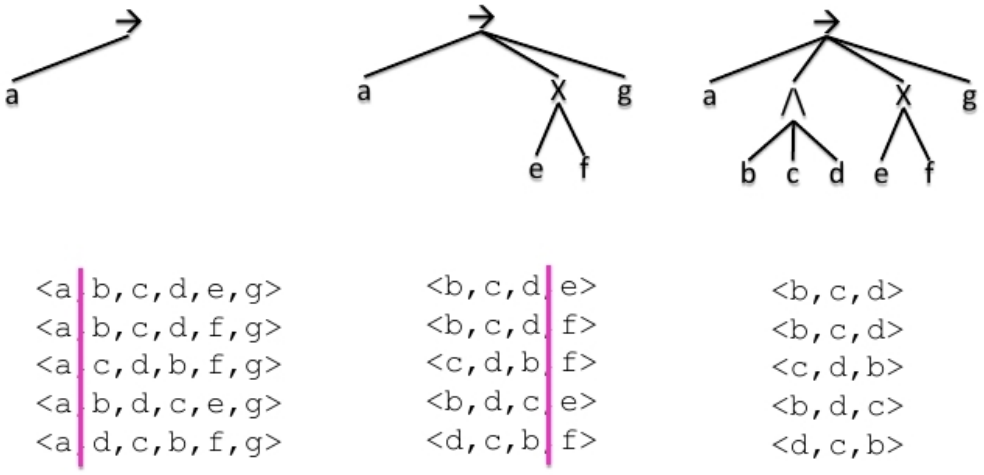
FIGURE 2.7: Dependency Matrix des Heuristic Miners[Bui17]

$$a => b = \frac{|a > b| - |b > a|}{|a > b| - |b > a| + 1}$$

It can be seen that the frequency of |a>b| - meaning a was followed by b - is 56. The opposite |b>a| never occurred and for this reason, by applying the formula above, the relative frequency or "significance of the dependency" of 0.98 can be calculated. The opposing relative frequency of -0.98 can be derived from the first result and is inserted in the matrix as well.[Bui17]

2.2.3 Inductive Miner (IM)

With the Inductive Miner the idea of classification by decision trees is introduced to process mining and because it doesn't use Petri nets, it guarantees sound process models. The IM repeatedly finds the most prominent splits in the event log, then detects the operator (e.g. X) and afterwards continues on both sublogs.
In the following Figure the example event log from this chapter is first split into the most prominent activities a (start) and g (end). Afterwards the OR operator is applied for e and f and finally b,c and d are split with and the parallel operator detected for this split.[Bui17]



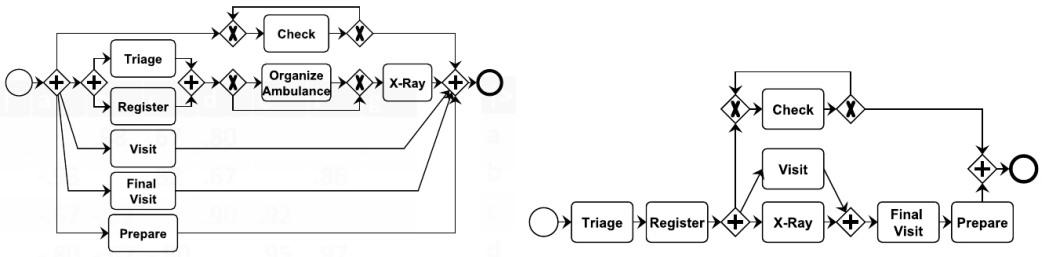
(a) Always a (b) Always e OR f (c) Parallelism b,c,d

FIGURE 2.8: Inductive Miner - Repeatedly Split Event Log[Bui17]

2.3 The Data-aware Heuristic Miner (DHM)

2.3.1 Data-aware Dependency Measures

In contrast to the miners which have been shortly presented in this chapter, the DHM is supposed to also consider and evaluate infrequent but data-dependent process behavior. Rare events are either vastly disregarded as noise e.g. by the HM or not properly filtered e.g. by the IM.



(a) Inductive Miner (b) Heuristic Miner

FIGURE 2.9: Discovered models by the IM and HM in BPMN [Man17]

The approach of Mannhardt et. al is to extend the HM with a measure for conditional dependency. Binary classifiers are applied to predict directly-follows relations based on attribute values recorded in the event log.
Those classifiers are denoted as **Dependency Conditions**:

$$C_{a,b}(x) = (C(a,b))(x)$$

This binary classifier predicts whether an event of activity a is directly followed by an event of activity b for the attribute values x. This means that $C_{a,b}(x) = 1$ when b is predicted to directly follow a and $C_{a,b}(x) = 0$ when a different activity is predicted.

Given $a, b \in \Sigma$ and dependency conditions C, the frequency with which b is observed to directly follow a is derived from the event log. This relation is denoted as:
Conditional directly follows relation

$$a >^{C,L} b$$

An execution of activity a with the latest attribute values x is directly followed by an execution of activity b under dependency condition $C_{a,b}(x)$.
The **Conditional dependency measure** is calculated analogous to the dependency measure described for the HM earlier. The authors define $a =>^{C,L} b : \Sigma \times \Sigma \rightarrow [-1, 1]$ as the strength of the causal dependency from a to b under condition $C_{a,b}$ in the event log:

$$a =>^{C,L} b = \begin{cases} \frac{|a>^{C,L}b| - |b>^{C,L}a|}{|a>^{C,L}b| - |b>^{C,L}a| + 1} & \text{for a} \neq b, \\ \frac{|a>^{C,L}b|}{|a>^{C,L}b| + 1} & \text{otherwise.} \end{cases}$$

The difference to the earlier measure is the data-aware dependency condition. If a relation (a, b) is clearly characterized by a certain dependency condition $C_{a,b}$ it should be included in the dependency relations of the discovered causal net.

The **emergency ward example** will provide a better understanding of the described measures. An event log L with 50 traces for each of the three σ_{1-3} was introduced earlier and will be used in the following steps:

1. Determine the conditional dependency measure $X =>^{C,L} V$ from activity X-Ray (X) to activity Visit (V)
   => Assumtion that condition $C_{X,V}(v) = 1$ only if attribute Nurse = Alice
2. Obtain the number of times X is directly followed by V under condition $C_{X,V}$
   => $|X >^{C,L} V|$ = 50
3. Obtain the number of times V is directly followed by X under conditions C
   => $|V >^{C,L} X|$ = 0
4. Derive the conditional dependency measure under C
   => $X =>^{C,L} V = \frac{50-0}{50+0+1} \approx 0.98$

This indicates a strong dependency relation from activity X to activity V under condition $C_{X,V}$. By contrast, if we consider the unconditional dependency measure $X =>^{1,L} V$, then we obtain $\frac{50-100}{50+100+1} \approx -0.33$. Thus, when disregarding the data perspective, both activities appear to be executed in parallel.[Man17]

2.3.2 Training a classifier for the dependency condition

The **dependency condition** which was introduced above, is the basis of the other two definitions and thus the first step when applying the DHM in order to decide which relations should be included in the C-net.
A set of training instances is built for every combination of activities $(a, b) \in \Sigma \times \Sigma$.
Training Instances are defined by:

- $a \in \Sigma$ the given source activity
- $b \in \Sigma$ the candidate activity
- $\theta_{dep} \in [0, 1]$ the dependency threshold

Then $a\bullet \subseteq \Sigma$ is the set of activities s that directly follow a in the event log with an unconditional dependency measure above the threshold θ_{dep} , i.e.,

$$a\bullet = s \in \Sigma | a =>^{1,L} s \geq \theta_{dep}$$

We collect those events $X_{L,a,b} \subseteq E$ that directly follow an execution of a in the event log, and refer to activities in $a\bullet$, or to the candidate activity b, i.e.,

$$X_{L,a,b} = e \in E | \bullet(e) = a \wedge \#_{act}(e) \in a\bullet \cup \{b\}$$

Function $T_{L,\theta_{dep}} \in (\Sigma \times \Sigma) \rightarrow B((A \nrightarrow U) \times 1, 0)$ returns the multi-set of training instances:

$$T_{L,\theta_{dep}}(a,b) = \biguplus_{e \in X_{L,a,b}} [(val(e), cl(e))] with cl(e) = \begin{cases} 1, & for \#_{act}(e) = b, \\ 0, & for \#_{act}(e) \neq b \end{cases}$$

This method is conceptually independent of the used classification algorithm. The authors employed **decision trees (C4.5)** as they are efficient and provide results in human interpretable conditions.
To build the **dependency conditions C**:

1. A set of training instances $T_{L,\theta_{dep}}(a,b)$ is assembled.
2. A decision tree for each possible relation $(a,b) \in \Sigma \times \Sigma$ is trained.
3. A score $q(C_{a,b}) \in [0,1]$ is used to determine the quality of a condition $C_{a,b}$.

As a performance measure the authors opted for Cohen's kappa (κ), which indicates whether the prediction was better than a prediction by chance (i.e., for κ > 0).

Again the **emergency ward example** with an event log of 150 traces will provide a better understanding of the just described methods.
The dependency threshold shall be $\theta_{dep} = 0.9$ and the classifier for the dependency condition $C_{X,V}$, i.e., the dependency relation from X-Ray (X) to Visit (V) will be trained:

1. The training instances are $T_{L,\theta_{dep}}(X,V) = [(v1, FinalVisit)^{50}, (v2, Visit)^{50}]$
2. The attribute value functions are $v1(P) = Red$, $v1(N) = Joe$ and $v2(P) = Red$, $v2(N) = Alice$
3. A C4.5 decision tree is trained and the dependency condition $C_{X,V}$ with $C_{X,V}(v2) = 1$ and $C_{X,V}(v1) = 0$ is obtained

There is no instance with the activity Check (C) since the unconditional dependency measure $X =>^{1,L} C$ is below the threshold of 0.9. So the instances based on trace $\sigma 3$ are not included because activity C is in parallel to X.[Man17]

2.3.3 Tuning noise filtering capabilities and discovering C-nets

In order to filter the noise from rare events the DHM supports four user-specified thresholds which range between 0 and 1:

- θ_{obs}, observation threshold - controlling the relative frequency of relations

- θ_{dep}, dependency threshold - controlling the strength of causal dependencies
- θ_{bin}, binding threshold - controlling the number of bindings
- θ_{con}, condition threshold - controlling the quality of data dependencies

A C-net tuple $C = (\Sigma, s_i, s_o, D, I, O)$ is derived from an event log $L = (E, \Sigma, \#, L)$ and the thresholds θ_{obs}, θ_{dep}, θ_{bin}, θ_{con}, in the following steps.

1. Add artificial start and end events to all traces to ensure unique start and end activities s_i and s_0
2. Build the set of standard dependency relations
3. Discover the dependency conditions C by training the classifiers for each pair (a, b), using the training instances $T_{L,\theta_{dep}}(a, b)$.
4. Add the conditional dependency relations C to D, using θ_{con} instead of θ_{obs} to obtain infrequent, high-quality data conditions
5. Handle activities $s \in \Sigma$ which don't have a predecessor or successor in the directed graph induced by D. As all tasks in the C-net should be connected, two alternative heuristics are proposed by the authors: **all-task-connected** and **accepted-task-connected**. The latter one connects repeatedly only those activities that are already part of the dependency graph using their best neighboring activities until all activities have a cause and an effect.
   $\bar{D}$ denotes the set of relations necessary to connect all activities accepted so far. Afterwards the dependency relations with the new relations, i.e., $D = D \cup \bar{D}$ is extended. As now there might be new, unconnected activities in $\bar{D}$ the steps are repeated i.e. the best neighboring activities are added until set $\bar{D}$ is empty.
6. The input and output binding functions of the C-net are discovered. We discover the input (I) and output (O) binding functions of the C-net. To find $O(a)$ it has to be determined which of the executions of b were caused by an execution of activity a. Using the same heuristic like before activity a only is considered to have caused b if it is the nearest activity. Any other activity s executed in between a and b should not be a possible cause of b. $\bar{O}$ denotes the set of activities that were caused by event e_i.

   The frequency $|o|_{L,a} \in N$ of an output binding $o \subseteq \Sigma$ for activity $a \in \Sigma$ in the event log L is determined as:

$$O(a) = \{o \subseteq \Sigma | \frac{|o|_{L,a}}{max_{\bar{o} \subseteq \Sigma}(|\bar{o}|_{L,a})} \geq \theta_{bin}\}$$

   The input binding function I is obtained by reversing the same approach.[Man17]

Chapter 3

Result Reproduction

3.1 Synthetic Data Logs

3.1.1 Evaluation design and methods

The authors generated an event log with 100,000 traces and approximately 900,000 events.

Attributes	Methods	Thresholds
• Priority (P) white 1.4% • Nurse (N) Alice 19.1% • Type (T) out 4.3%	• DHM • HM - frequency filter (HMF) • HM - no frequency filter (HMA)	• θ_{obs} = 0.06 (0.0 for HMA) • θ_{dep} = 0.9 • θ_{bin} = 0.1 • θ_{con} = 0.5

Furthermore the **accepted-task-connected heuristic** was used, the **C4.5** applied as the classifier and its performance was estimated using **10 times 10-fold cross validation**.

3.2 Hospital Billing

The Hospi- tal Billing (HB) event log was obtained from the ERP system of a hospital. It contains 100,000 cases with 550,000 events and 38 data attributes related to the billing of medical services.

CaseType CloseCode isClosed Speciality

3.3 Road Fines

The Road Fines (RF) event log was recorded by an information system that handles road-traffic fines by an Italian local police force. This event log contains about 150,000 cases, 500,000 events, and 9 data attributes.

We applied the proposed method (DHM), the HM with the same frequency filter settings (HMF) and the Inductive Miner (IM) to both event logs. Without a reference model and knowledge about expected noise levels, we could not compare the

discov- ered models to a gold standard. Therefore, we compare the novel insights obtained by using our method with those from the other methods.

Bibliography

[Bui17] Joos Buijs. *Introduction to Process Mining with ProM*. 2017. URL: https://www.futurelearn.com/courses/process-mining?utm%7B%5C_%7Dcampaign=eindhoven%7B%5C_%7Duniversity%7B%5C_%7Dof%7B%5C_%7Dtechnology%7B%5C_%7Dprocess%7B%5C_%7Dmining%7B%5C_%7Dseptember%7B%5C_%7D2017%7B%5C&%7Dutm%7B%5C_%7Dmedium=organic%7B%5C_%7Demail%7B%5C&%7Dutm%7B%5C_%7Dsource=newsletter%7B%5C_%7Dbroadcast.

[Man17] et. al. Mannhardt Felix. "Data-driven process discovery-revealing conditional infrequent behavior from event logs". In: *International Conference on Advanced Information Systems Engineering - Springer, Cham, 2017* (2017). URL: https://link.springer.com/chapter/10.1007/978-3-319-59536-8%7B%5C_%7D34.

[Van+11] Wil Van Der Aalst et al. "Causal nets: a modeling language tailored towards process discover". In: *International conference on concurrency theory-Springer* 1 (2011). URL: https://link.springer.com/10.1007%7B%5C%%7D2F978-3-642-23217-6%7B%5C_%7D3.