Improving the quality of the Heuristics Miner in ProM 6.2.

Sofie De Cnudde^{a,1,*}, Jan Claes^a, Geert Poels^a

^a*Department of Business Informatics and Operations Research, Faculty of Economics and Business Administration, Ghent University, Tweeckerkenstraat 2, B-9000 Ghent, Belgium*

Abstract

Considering the presence of large amounts of data in organizations today, the need to transform this data into useful information and subsequently into knowledge, increasingly gains attention. Process discovery is a technique to automatically discover process models from data in event logs. Since process discovery is gaining attention among researchers as well as practitioners, the quality of the resulting process model must be assured. In this paper, the quality of the frequently used Heuristics Miner is improved as anomalies were found concerning the validity and completeness of the resulting process model. For this purpose, a new artifact called the Updated Heuristics Miner was constructed containing alterations to the tool and to the algorithm itself. Evaluations of this artifact resulted in the conclusion that the Updated Heuristics Miner indeed demonstrates higher validity and completeness. This study contributes to the body of knowledge first by improving the quality of the an often used research instrument and second by stating that there is a need for a systematic developing and evaluation method for process discovery techniques.

Keywords: knowledge discovery, process mining, Heuristics Miner, quality, validity, completeness

1. Introduction

The quest for transforming big amounts of data into information and subsequently into knowledge, is a widely researched topic due to the presence of such vast amounts of data in organizations today. The technique of process mining constitutes one of these attempts by extracting information about business processes from data
5 that is gathered through information systems in an organization.

Process mining can be subdivided into three types: process discovery, conformance checking and process enhancement (Van der Aalst (2011)). Process discovery makes an attempt at generating a process model from the process execution in the organization. Conformance checking techniques check if the actual execution conforms
10 to the designed process execution. Lastly, process enhancement techniques try to enrich a process model with extra information such as process execution information.

*Corresponding author. Tel: +32 3 265 50 14

Email addresses: sofie.decnudde@uantwerp.be (Sofie De Cnudde), jan.claes@ugent.be (Jan Claes), geert.poels@ugent.be (Geert Poels)

¹Present address: Department of Engineering Management, Prinsstraat 13, B-2000 Antwerp

The technique of process discovery is of practical relevance for organizations, since it can give insights into the behavior of their processes. These insights can consequently lead to the optimization of their (critical) processes, which can ultimately lead to increased profits or decreased losses.

15 Process discovery techniques, and more specifically the Heuristics Miner technique, is being used in several organizations of which, among others, the following case studies provide proof (Jans et al. (2011), Mans et al. (2009), Rozinat et al. (2007a), Rebuge & Ferreira (2012), Engel et al. (2012a), Wang et al. (2012)). Since it is no longer only an academic artifact, it is important that the algorithm and the heuristics used in the Heuristics Miner technique are reliable and thus provide valid and complete results for organizations. However, when
20 performing the Heuristics Miner algorithm, as implemented in ProM 6.2, on an event log from the Belgian transportation company *De Lijn*, two anomalies were found. Firstly, split and join semantics were not correctly mined and shown in the discovered process model. Second, relations constituting approximately 44 % of the entire event log were not present in the mined process model. Considering the Heuristics Miner algorithm is popular in research and in practice (as will be shown in Section 3), for the method not being able to mine nearly
25 50% of the behavior present in the data is not acceptable.

To our knowledge, these anomalies relating to the validity and the completeness of the mined process models in the open source process mining framework ProM have not yet been resolved. The goal of this research is to develop an artifact (cfr. the design science research methodology described in Peffers et al. (2007)) based on the
30 Heuristics Miner algorithm, but not containing the deficiencies concerning validity and completeness.

The rest of this work is organized as follows. In the following section, three general elements of process mining are first described, i.e. process discovery in general, the Heuristics Miner technique and evaluation metrics for process discovery techniques. Next, in Section 3 important notions concerning the research methodology are
35 described, including arguments in favor of the relevance of this work and the objectives of this paper. Section 4 and Section 5 describe the structural problems and the implementation problems in the Heuristics Miner respectively. In Section 6, an attempt is made to evaluate our outlined artifact which we call the Updated Heuristics Miner. This section also presents possible threats to the validity of our evaluation. Finally, Section 7 concludes our work, states the main contributions of our work, along with opportunities for further research.

40 2. Background

2.1. Process discovery

The input of a process discovery technique is an event log. An event log consists of a sequence of traces, which, in turn, consist of a sequence of events that were recorded in the information systems of an organization. We describe an event log W_1 as follows: $\{ABCDEF^3, AGHF^2, ABDCEF^2\}$, where the superscripts denote the
45 frequency of the trace in the event log.

According to van der Aalst et al. (2007), the information system must be able to record events such that: “(i)

each event refers to an activity (i.e. a well-defined step in the process), (ii) each event refers to a case (i.e. a process instance, meaning a particular execution of the process), (iii) each event can have a performer or an originator (i.e. the actor executing or initiating the activity) and (iv) each event has a timestamp so that all events are totally ordered.”

Process discovery algorithms generate output, i.e. information, from the data in the event log. There are three different perspectives from which an event log can be mined and which subsequently produce different output (van der Aalst et al. (2007)): the process (“How?”), the organizational (“Who?”) and the case (“What?”) perspective. The first perspective shows the order in which the activities are performed in a process model. The Heuristics Miner mines the event log following this perspective. For event log W_1 , the process model in Figure 1 could be the output of some process perspective discovery technique. The organizational perspective gives a social network of the different roles and/or relations in an organization. The case perspective, at last, “focuses on properties of the cases” (van der Aalst et al. (2007)). The process and the organizational perspective are thus specific cases of the case perspective, using the path in the process and the originator, respectively.

An event log may contain noise. As defined in van der Aalst et al. (2007), noise can refer to parts of the log (i.e. traces or activities) which (i) refer to exceptions, (ii) were recorded incorrectly or (iii) are incomplete. Considering the possible presence of noise in an event log, process discovery techniques must make a trade-off between overfitting and generalization, i.e. including every recorded trace of events versus leaving room for traces not captured in the event log, but which may occur in practice.

2.2. Heuristics Miner

The Heuristics Miner mines an event log in three phases: (i) mining the dependency graph, (ii) mining split and join relations, (iii) mining long-distance dependency relations. Step (iii) is not relevant for our research, and thus will not be discussed in more detail here. We refer to Weijters & Ribeiro (2011) for a description of this step.

Figure 1: A possible mined process model of event log W_1 .

The goal of the first phase consists of building the dependency graph related to the event log. As mentioned in Weijters & Ribeiro (2011) the dependency graph of an event log is defined as follows:

$$DG = \{(a, b) | (a \in E \wedge b \in a\Box) \vee (b \in E \wedge a \in \Box b)\}, \quad (1)$$

where E is the finite set of activities, for which events are recorded in the event log, $\Box b$ denotes the activities preceding b , and $a\Box$ consists of the activities succeeding a . In other words, the dependency graph contains for each activity a the input- and output activities that have been accepted as dependency relations (a, b) .

In order to accept a dependency relation and consequently build the dependency graph of an event log, first a dependency matrix is constructed. The dependency matrix of an event log W is a $n \times n$ matrix for which each element $d_{a,b}$ is defined as follows:

$$d_{a,b} = a \Rightarrow_W b = \left(\frac{|a >_W b| - |b >_W a|}{|a >_W b| + |b >_W a| + 1} \right), \quad (2)$$

with n the number of activities in W , a and b two different activities present in event log W and $|a >_W b|$ the frequency of $\dots ab \dots$ sequences in event log W . The value $a \Rightarrow_W b$ is called the dependency measure between two activities a and b and is a frequency-based metric that quantifies the certainty of a dependency relation between two activities a and b . It can take on a value between -1 and 1 : a high value denoting a strong dependency, a low value denoting a strong inverse dependency, and a value of zero denoting no dependency.

Since Definition 2 results in very low values for loops of length one (i.e. ACB , $ACCB$, $ACCCB$, ...) and loops of length two (i.e. $ACBD$, $ACBCBD$, $ACBCBCBD$, ...) due to the low value in the nominator, the Heuristics Miner algorithm (Weijters & Ribeiro (2011)) provides an alternative definition of the dependency measure for loops of length one and length two (which are called L1 loops and L2 loops respectively):

$$a \Rightarrow_W a = \left(\frac{|a >_W a|}{|a >_W a| + 1} \right), \quad (3)$$

$$a \Rightarrow_W^2 b = \left(\frac{|a >>_W b| + |b >>_W a|}{|a >>_W b| + |b >>_W a| + 1} \right), \quad (4)$$

75 with $|a >>_W b|$ the frequency of $\dots aba \dots$ sequences in event log W . The dependency measure for these short loops of length one and two are defined in a range between 0 and 1.

The dependency matrix contains for each two activities the strength of their connection. The goal now is to select the strongest connections between events that will be shown in the resulting mined process model. This selection is done using thresholds and optionally the all-tasks-connected heuristic. The Heuristics Miner defines
80 five thresholds: the dependency threshold, the length-one loops threshold, the length-two loops threshold, the relative to best threshold and the positive observation threshold which are used to a different extent dependent on the use of the all-tasks-connected heuristic.

In case of constructing the dependency graph without the all-tasks-connected heuristic, threshold values are used to determine if the value of the dependency measure is high enough and consequently if the dependency relation will be accepted or not. Three threshold values are used: the dependency threshold, the length-one loops threshold and the length-two loops threshold. An advantage of using these three threshold values is that an experienced user can set them in such a way that only the very strong dependency relations are shown (i.e. setting a high threshold). The disadvantage, however, is that a user must have some sort of knowledge concerning the process, the discovery algorithm or the purpose of the resulting process model.

The all-tasks-connected heuristic somewhat tries to eliminate this disadvantage. It uses the following intuition: each activity x , that does not start or end a trace, must have an activity a that triggers its execution and an activity b that is triggered by its execution, i.e. all activities are connected. Using this heuristic, a preliminary dependency graph is built: for each activity x the input set $\Box x$ consists of the activity a having the highest value for the dependency measure $a \Rightarrow_W x$, and analogous for the output set $x \Box$. Extra dependency connections are accepted with the use of the relative to best threshold in combination with the dependency threshold. If a dependency measure $a \Rightarrow_W y$ has a higher value than the dependency threshold and is “close enough” to the highest value of the dependency measure $a \Rightarrow_W x$, i.e. the difference does not exceed the relative to best threshold, then the dependency relation $a \Rightarrow_W y$ is also accepted.

One extra threshold value is used in the Heuristics Miner, i.e. the positive observation threshold, which sets a lower bound to the frequency of an $\dots ab \dots$ sequence in the event log. This threshold is default set to 1.

Not using the all-tasks-connected heuristic may result in a dependency graph with disconnected activities, yet connectedness seems a desirable process model property. Therefore, the assumption is made that the all-tasks-connected heuristic is always used (the option is also set default in ProM 6.2).

2.2.2. Mining split and join relations

Figure 2 shows the process model related to event log W_1 , enriched with split and join semantics. We call such process models enriched with non-sequential process flow information semantic process models. Note that the symbol for an AND-split and an AND-join is the $\oplus$ -symbol and the symbol for an XOR-split and an XOR-join is the $\otimes$ -symbol.

In order to construct a so-called semantic process model, first the dependency graph is transformed into a causal net. A causal net (A, I, O) is defined as follows (Weijters & Ribeiro (2011)):

$$A \text{ is a finite set of activities,} \quad (5)$$

$$I : A \rightarrow \mathcal{P}(\mathcal{P}(A)) \text{ is the input pattern function,} \quad (6)$$

$$O : A \rightarrow \mathcal{P}(\mathcal{P}(A)) \text{ is the output pattern function.} \quad (7)$$

The transformation from $\Box a$ to $I(a)$ and from $a \Box$ to $O(a)$ for an activity a can be done as follows. First, the following metric is determined for each two elements in the input or output set of an activity a in an event log

Figure 2: A possible mined process model of the event log W_1 , enriched with semantic split and join information (semantic process model).

W (Weijters et al. (2006)):

$$a \Rightarrow_W b \wedge c = \left(\frac{|b >_W c| + |c >_W b|}{|a >_W b| + |a >_W c| + 1} \right). \quad (8)$$

This metric is high when traces $\dots abc\dots$ and $\dots acb\dots$ often occur in an event log, which means that activities b and c are in an AND-relation, with respect to activity a . The value of this metric is then compared to a threshold value, which is default set to 0.1. If the value is higher than this threshold value, then the activities b and c are assumed to be in an AND-relation, otherwise they are assumed to be in an XOR-relation. Finally, the split and join semantics are stored in the causal net. For the complete dependency graph of event log W_1 , the causal net is shown in Table 1. Note that elements in the same subset are in an AND-relation and elements in different subsets are in an XOR-relation.

2.3. Evaluating the quality of the discovered process models

The process discovery techniques need to be evaluated in terms of their generated output. This evaluation may take place alongside the following three dimensions (Krogstie et al. (1995)): syntactic quality, pragmatic quality and semantic quality. Syntactic quality refers to the correspondence between the process model and the syntax of the process modeling language. The pragmatic quality refers to the comprehension of the process model. These two dimensions are not subject of our work and we will further focus only on semantic quality. We thus use the semiotic quality framework of Krogstie et al. (1995) to define quality here as the semantic quality of a model. The semantic quality consists of validity and completeness. Validity refers to the fact that all statements made by the process model are correct, completeness means that all relevant behavior is present in the mined process model.

A common framework for evaluating mined process models is not yet present in the process mining literature. However, Rozinat et al. (2007b) and Rozinat et al. (2008) presented a first attempt to this end. They stated the need for comparing mined process models resulting from different algorithms and, more importantly, the need to assess the quality of mined process models. The following dimensions for the evaluation of the quality of process models were defined by Rozinat et al. (2007b) and Rozinat et al. (2008): (i) fitness, (ii) precision, (iii) generalization and (iv) structure. Fitness is defined as the “fit” of the process model with the “reality”, i.e. the

INPUT SET	ACTIVITY	OUTPUT SET
$\{\emptyset\}$	A	$\{\{B\}, \{G\}\}$
$\{\{A\}\}$	B	$\{\{C, D\}\}$
$\{\{B\}\}$	C	$\{\{E\}\}$
$\{\{B\}\}$	D	$\{\{E\}\}$
$\{\{C, D\}\}$	E	$\{\{F\}\}$
$\{\{E\}, \{H\}\}$	F	$\{\emptyset\}$
$\{\{A\}\}$	G	$\{\{H\}\}$
$\{\{G\}\}$	H	$\{\{F\}\}$

Table 1: Causal net of event log W_1 .

observed behavior recorded in the event log. The fitness of a process model is poor if an event log contains a lot of traces which can not be parsed by the process model. Precision penalizes overfitted models, generalization penalizes underfitted models and structure is related to the pragmatic quality of a model.

Another attempt at creating a common framework for the evaluation of mined process models can be found
135 in De Weerd et al. (2012). Here, recall (the percentage of behavior captured by the process model), precision (a measure for the trade-off made between overfitting and underfitting the model) and comprehensibility (the understandability of the process model) are viewed as the main dimensions of process model quality.

Since the latter is related to the pragmatic quality of a model, it is not included in our study. The fitness
140 measure is the evaluation measure for recall we will use in this paper. Concerning the precision dimension, this will be visually assessed by the researchers since measures for precision are computationally complex and no straightforward and precise measure can be found in the ProM 6.2 framework (De Weerd et al. (2012)).

Two metrics were found in the Heuristics Mining literature which quantify fitness (Weijters et al. (2006)): the parsing measure PM and the continuous parsing measure CPM . The parsing measure is defined as follows:

$$PM_W = \frac{c}{t}, \quad (9)$$

with c the number of correctly parsed traces and t the total number of traces in event log W . A second, less naive (Weijters et al. (2006)) continuous parsing fitness measure is defined with a higher granularity (i.e. identifying events instead of traces) as follows:

$$CPM_W = \frac{1}{2} \frac{(e - m)}{e} + \frac{1}{2} \frac{(e - r)}{e}, \quad (10)$$

with e the total number of events in event log W , m the number of missing activated input expressions (i.e. events not parsed in the trace) and r the number of remaining activated output expressions (i.e. the hanging
145 events at the end of a trace).

3. Research Methodology

In designing the Updated Heuristics Miner artifact, the design science research methodology of Peffers et al. (2007) was followed. The design science research methodology consists of the following steps: problem identification and motivation, objectives of the artifact, design and development, demonstration and evaluation. The initial motivation for the Updated Heuristics Miner artifact consisted of a concrete problem found in a specific event log. When performing the Heuristics Miner in ProM 6.2 on a customer-complaint handling event log W_{RL} of the Belgian transportation company *De Lijn*, it was shown that the generated semantic process model did not comply with theoretical foundations. Moreover, nearly 50% of the behavior was not present in the mined process model. The goal of the research then consisted of using the design science research methodology (DSRM) to solve these anomalies. While in the development phase of this first cycle of the DSRM, other problems were found concerning the Heuristics Miner algorithm and the implementation in ProM 6.2 (which will be referred to here respectively as structural and implementation problems) which did not only affect the specific event log W_{RL} , but also other real-life event logs. Consequently, a second cycle of the DSRM was started, initiating from the validity and the completeness of the Heuristics Miner in general and not related to any event log in particular. This benefited our research in terms of the generalization of the designed artifact, i.e. the Heuristics Miner was not optimized to a particular event log.

This paper will set out the second cycle of the DSRM, using the problematic event log W_{RL} as an extra means to evaluate our Updated Heuristics Miner (UHM).

3.1. Relevance

To demonstrate the relevance of this paper, the following references are made. First of all, Van der Aalst & Weijters (2004) discussed challenging problems in the field of process discovery. Two of these challenges concern the mining of loops and the presence of noise. These two challenges are related to the structural problems as we will discuss in Section 4. We will tackle these challenges and our artifact will be shown to better handle the mining of loops and the treatment of noise. Second, the process mining manifesto (Van der Aalst et al. (2012)) mentioned three challenges that contribute to the relevance of this study. Challenge C2 states that process discovery techniques should be able to deal with different sorts of event logs with different characteristics. The event log from the transportation company *De Lijn* indeed contains traces with sequences of events not encountered in other available event logs. The developed artifact will thus contribute to the ability of the Heuristics Miner to mine a wider range of event logs. Challenges C10 and C11 state that the usability and understandability for non-experts must be improved. By improving the quality of the discovered process models, they become more reliable and users will be able to use them more frequently for e.g. decision making. The understandability of the discovered process model is a prerequisite for usability, and solving problems related to semantic validity contributes to a more understandable process model. Next, the survey conducted by Claes & Poels (2013) stated that “the ProM framework is the most popular process mining tool for researchers and practitioners” and that the Heuristics Miner is a frequently-used tool to mine event logs. It is also perceived

as a fast, and fairly intuitive, understandable and trustworthy process mining technique. Thus, it is important that the Heuristics Miner generates reliable results. This survey also puts forward that researchers also use the Heuristics Miner. This contributes to the research relevance of our study. Fourth, case studies were found that use the Heuristics Miner to mine real-life event logs: Mans et al. (2009), Rozinat et al. (2007a), Rebuge
185 & Ferreira (2012), Engel et al. (2012a), LANGab et al. (2008), Mans et al. (2008), Engel et al. (2012b), Fei & Meskens (2008), Wang et al. (2014), van der Aalst et al. (2007). In these case studies, conclusions concerning the Heuristics Miner and its quality were made based on the discovered process model and thus may be influenced by the fact that some important connections are not present in the mined process model. Moreover, further research might be based on these faulty conclusions. Making faulty conclusions or, in a business context, faulty
190 decisions, is not desirable and also proves the relevance of this study. Lastly, Gupta (2007) stated that when mining an event log with the Heuristics Miner, many connections were not shown in the discovered process model.

3.2. Objectives

Following the design science research methodology, we now define three clear objectives for the designed artifact.
195 We state that the designed artifact must on the one hand generate valid process models, i.e. the process models must be correct in line with the theory concerning the Heuristics Miner (Weijters & Ribeiro (2011)). On the other hand, the Updated Heuristics Miner must also generate complete process models. A complete process model is not easily definable since a trade-off must be made between overfitting and underfitting the reality present in the event log. In other words, the Updated Heuristics Miner must mine process models with a high
200 fitness value (high fit with reality), while not overfitting the event log by including noisy connections. Moreover, these objectives must be achieved while not increasing the execution time or increasing the effort for users.

4. Structural problems in the Heuristics Miner algorithm

Three structural problems were found in the algorithm of the Heuristics Miner and deteriorated the semantic quality of mined process models. One problem concerned the validity of the mined process models and two
205 problems concerned the completeness of the mined process models.

4.1. Validity of the mined process model

The Heuristics Miner algorithm defines separate dependency measures for loops of length one and loops of length two (L1 and L2 loops), as shown in Definition 3 and Definition 4. These formulas seem somewhat counterintuitive when considering the following illustration. The presence of 9 L1 loops in an event log of length 1,000,000,
210 results in the acceptance of that loop, assuming that the L1 loop threshold equals the default value of 0.9. Yet, these 9 occurrences must, in our opinion, be considered noise and thus not be shown in the resulting process model. Analogous for L2 loops, an event log with 1,000,000 traces containing 5 traces with ...ABAB... or only one trace with ...ABABABABABAB..., results in a process model containing the L2 dependency connection

between activities A and B .

215 This problem is associated with the fact that there is no noise defined for loops of length one or two. The dependency measure $a \Rightarrow_W b$ between two different activities considers the occurrence of $b >_W a$ as noise to the relation $a \Rightarrow_W b$ and thus penalizes the $a \Rightarrow_W b$ value in the form of a lower dependency measure. In contrast, these loop dependency measures are never penalized.

220 Since validity refers to the fact that all statements made by the process model must be correct and relevant, this structural problem amounts to a lower validity of the discovered process model. Showing noise in the process model is not relevant and does not amount to a valid model.

Investigating Definition 3 and Definition 4 in more detail, it is clear that they merely are count functions of the occurrence of these loops. The numerator and denominator always differ only 1 unit, so that $a \Rightarrow_W a$ and $a \Rightarrow_W^2 b$ can only take on values of the following form: $\frac{n}{n+1}$, with n a natural number. Consequently, the value for the dependency measures starts from $\frac{1}{2}$ and only has 8 possibilities (i.e. from $\frac{1}{2}$ to $\frac{8}{9}$) of taking on a value lower than 0.9, which is the default threshold value for loops of length one and loops of length two. There are thus only 9 cases in which a short loop will not be shown, assuming the default threshold value.

Even when assuming that the default threshold value for L1 and L2 loops is not used, then no meaningful threshold value can be chosen by the user, since the possible values for the dependency measures are not equally spread across the $[0, 1]$ -interval, i.e. all but 8 possible values for the dependency measures are located in the $[0.9, 1]$ -interval. A possible solution could be to provide a transformation from the $[0.9, 1]$ -interval to the $[0, 1]$ -interval, but this does not alter the faulty short loops-measure. Therefore, another definition for the dependency measures for L1 and L2 loops is defined.

An attempt is made to define a dependency measure for L1 loops in accordance to the definition of the dependency measure for ordinary dependency connections, i.e. we wish to incorporate a penalty for noise for the $a \Rightarrow_W a$ connection. First, noise has to be defined with respect to L1 loops, but this is not straightforward. For normal dependency relations $a \Rightarrow_W b$, the “opposite relation” $b \Rightarrow_W a$ is considered noise. Here however, this is not possible since we are considering loops of length one. An option that was explored was considering all relations $a \Rightarrow_W x$ (with $x \neq a$) as noise for $a \Rightarrow_W a$. However, this assumption leads to an exaggeration of the noise and is not accurate. A golden mean is hard to find between these two extremes, since considering some $a \Rightarrow_W x$ connections as noise and some not, leads to a distorted dependency measure.

Therefore, a relative frequency-based measure is defined for L1 loops as follows:

$$a \Rightarrow_W a = \frac{|a >_W a|}{\max\{|a >_W x| | x \in e\}}, \quad (11)$$

with W an event log and e the activities in event log W .

The measure takes the ratio of the frequency of the L1 loop and the frequency of the relation between the activity and its best output activity. In other words, it expresses the strength of the L1 loop against the best connection the activity in the loop has. This results in a value between 0 and 1 that represents the strength of

the loop connection.

Analogous to short loops of length one, noise was attempted to be defined for L2 loops. However, also, the “opposite” relation $|b \gg_W a|$ can not be considered noise for the $a \Rightarrow_W^2 b$ connection, since $|a \gg_W b|$ and $|b \gg_W a|$ both denote an L2 dependency. Considering all relations $a \Rightarrow_W x$ (with $x \neq b$) as noise to a dependency connection $a \Rightarrow_W^2 b$ also leads to an exaggeration of noise for the L2 loop and is not accurate.

A new dependency measure must first take into account that there are two aspects to an L2 relation, i.e. the connection between a and b and the connection between b and a . If they both occur more in an L2 loop then the positive observation threshold and one connection $|a >_W b|$ or $|b >_W a|$ is strong enough, than the opposite connection must be made and the L2 loop must be recognized. Therefore, the dependency values for $a \Rightarrow_W^2 b$ and $b \Rightarrow_W^2 a$ must be equal. A dependency measure based on the relative frequency of the occurrence of the L2 loop is defined as follows:

$$a \Rightarrow_W^2 b = \max\left(\frac{|a >_W b|}{\max\{|a >_W x| | x \in e \wedge x \neq b\}}, \frac{|b >_W a|}{\max\{|b >_W x| | x \in e \wedge x \neq a\}}\right), \quad (12)$$

with W an event log and e all the activities in event log W .

225 It is clear that the measure produces the same value for both $a \Rightarrow_W^2 b$ and $b \Rightarrow_W^2 a$. The measure thus represents, analogous to L1 loops, the strength of an L2 loop dependent on the best connection an activity in the loop has.

4.2. Completeness of the mined process model

For regular dependency connections between two activities a and b , the Heuristics Miner (using the all-tasks-connected heuristic) connects all tasks based on their best input and output measure and then, to allow other
 230 strong dependency connections with a value close to the best connection, the relative to best threshold and the dependency threshold are used to also accept those strong connections. What seems counterintuitive about the use of the relative to best threshold, is that it is used in combination with the ordinary dependency threshold. In other words, consider an event log W_2 containing following traces $\dots ab\dots$ and $\dots ac\dots$ and $a \Rightarrow_{W_2} b = 0.61$, $a \Rightarrow_{W_2} c = 0.60$ and the all-tasks-connected heuristic is applied, then $a \Rightarrow_{W_2} b$ is accepted, but $a \Rightarrow_{W_2} c$ is
 235 not. The reason for this is that b is the best output event for activity a and is thus accepted, but when later applying the relative to best threshold, c is not accepted as an output activity for a because $a \Rightarrow_{W_2} c < 0.9$, i.e. lower than the dependency threshold. The value of $a \Rightarrow_{W_2} c$, however does not differ much from the best output measure of activity a and is thus as important to activity a as the $a \Rightarrow_{W_2} b$ -connection is. It does not seem a very true representation of the real world, if two dependency relations are almost equally strong, but only one
 240 is shown to the user. This might lead to wrong conclusions and thus to faulty decision-making.

This results in a lower completeness of the discovered process model and thus the use of the relative to best threshold is adapted in the Updated Heuristics Miner algorithm. The acceptance of extra dependency relations will now only be based on the relative to best threshold and the positive observation threshold.

245 A second encountered anomaly concerning completeness, concerns the use of the all-tasks-connected heuristic. This heuristic follows the intuition that, except for the first and last event, each event has a predecessor event and

a successor event and takes the events with the best dependency measures as these predecessor and successor activities. We have extended this heuristic to also include loops. It is stated in Weijters & Ribeiro (2011) that the use of the all-tasks-connected heuristic provides some advantages, among which, to a large extent, the independence of the three threshold-values: the regular dependency threshold, the length-one-loop dependency threshold and the length-two-loop dependency threshold. Consequently, we want to apply this heuristic not only to ordinary dependency relations, but also to L1 and L2 loops. This results in short loops now being accepted dependent on whether their connection to an activity is strong, rather than based on static dependency thresholds. This amounts to the completeness of the mined process model in terms of showing all relevant behavior.

5. Implementation problems in the Heuristics Miner in ProM 6.2

Besides the three structural problems described in Section 4, also two problems were found concerning the implementation of the Heuristics Miner in ProM 6.2. These anomalies deteriorated the semantic quality of process models generated using the Heuristics Miner in ProM 6.2. One problem concerned the validity of the mined process models and the other problem concerned the completeness of the mined process models.

5.1. Validity of the semantic process model

The semantic process model generated by the Heuristics Miner in ProM 6.2 is not considered valid. We define valid here as: it does not conform to the process model generated by the theoretical rules defined in Weijters & Ribeiro (2011), which describes the transformation from a dependency graph to a semantic process model.

When investigating in detail the program code responsible for building a semantic process model, two steps can be distinguished. The first transforms the dependency graph into a causal net and the second transforms the causal net into a semantic process model.

From dependency graph to causal net

Consider event log W_1 and its corresponding causal net in Table 1. When mining W_1 , the Heuristics Miner algorithm in ProM transforms the dependency graph shown in Table 2 into the causal net in Table 3. The dependency graph constructed by the Heuristics Miner algorithm is built correctly. However, when taking a closer look at the generated causal net, it is clear that this does not correspond to the causal net represented in Table 1, which was built according to the theory. Each activity in the causal net in Table 3 contains its correct input and output activities, but the construction of the subsets is not correct: two activities in the same subset are in an AND-relation and activities in different subsets are in an XOR-relation (Weijters & Ribeiro (2011)). It is clear that Table 3 contains the subsets in an opposite fashion.

From causal net to semantic process model

First, the activities itself are added to the semantic process model. Second, the input and output gateways for

INPUT SET	ACTIVITY	OUTPUT SET
$\{\emptyset\}$	A	$\{B, G\}$
$\{A\}$	B	$\{C, D\}$
$\{B\}$	C	$\{E\}$
$\{B\}$	D	$\{E\}$
$\{C, D\}$	E	$\{F\}$
$\{E, H\}$	F	$\{\emptyset\}$
$\{A\}$	G	$\{H\}$
$\{G\}$	H	$\{F\}$

Table 2: Dependency graph of event log W_1 generated by the Heuristics Miner in ProM 6.2.

INPUT SET	ACTIVITY	OUTPUT SET
$\{\emptyset\}$	A	$\{\{B, G\}\}$
$\{\{A\}\}$	B	$\{\{C\}, \{D\}\}$
$\{\{B\}\}$	C	$\{\{E\}\}$
$\{\{B\}\}$	D	$\{\{E\}\}$
$\{\{C\}, \{D\}\}$	E	$\{\{F\}\}$
$\{\{E, H\}\}$	F	$\{\emptyset\}$
$\{\{A\}\}$	G	$\{\{H\}\}$
$\{\{G\}\}$	H	$\{\{F\}\}$

Table 3: Causal net of event log W_1 generated by the Heuristics Miner in ProM 6.2.

each activity a are added. When adding a gateway, automatically an XOR-gateway is selected regardless of the nature of the relation between two activities. Then, an XOR-gateway is added for each input and output activity of a . Clearly this is incorrect, since an activity having several possible inputs (or outputs) should have only one XOR-gateway (or AND-gateway), as opposed to as many as its number of input (or output) activities.

285 Furthermore, since the erratic causal net representation is used, the algorithm considers an XOR-relation as an AND-relation and vice versa.

The third part of this transformation step, finally, connects the gateways to each other in order to form a connected semantic process model. The resulting semantic process model of W_1 is shown in Figure 3.

290 The source of this anomaly is thus twofold: an erratic representation of a causal net and the fact that the nature of a relation is not taken into account when transforming the causal net into a semantic process model. The semantic process model of event log W_1 shown in Figure 3 clearly shows no similarity to the correct semantic process model in Figure 2.

295 In order to solve this anomaly, first, the transformation from dependency graph to causal net is altered by correctly interpreting the AND- and XOR-relations from the input and output sets in the dependency graph. Then, the algorithm is adapted in order to make a distinction between activities in a sequential relation, an AND- or an XOR-relation. Dependent on the nature of the relation, the corresponding gateway is selected and added. Finally, the corresponding gateways are connected to each other. Figure 2 shows the semantic process model generated by the Updated Heuristics Miner for event log W_1 .

300

5.2. Completeness of the mined process model

The completeness of the mined process model concerns the fact that important dependencies are not shown. The source of this problem concerns the calculation of the L2 dependency measure.

Whereas in Section 4, we discussed the erroneous calculation of the L2 dependency measure, it was also observed that this measure was not always calculated to start with. More specific, the L2 dependency measure between two activities a and b is not calculated if one of the following conditions holds:

305

- a is in an L1 loop and $|a \gg_W b|$ is higher than the positive observation threshold or,

Figure 3: Screenshot of the semantic process model of event log W_1 generated with the Heuristics Miner in ProM 6.2.

- b is in an L1 loop and $|b \gg_W a|$ is higher than the positive observation threshold or,
- all of the above.

310 In other words, if one of two activities a or b is involved in an L1 loop and this activity occurs more than once in the L2 connection, then the Heuristics Miner algorithm will not accept this L2 relation. This is odd, since in that case, no activities involved in both a high-frequency length-one loop and a high-frequency length-two loop are shown in the mined process model.

This counter-intuitive condition is thus removed in the Updated Heuristics Miner.

315 6. Evaluation

The Updated Heuristics Miner is now evaluated in terms of validity and completeness on the one hand and in terms of cost, effort and time on the other.

6.1. Evaluating the validity of the mined process model

Validity is related to the dependency measures for short loops and the correctness of the semantic process models.

320

The goal of this first evaluation is to ascertain that a noisy L1 loop (or L2 loop) is no longer shown in the mined process model. The fitness measure is irrelevant here, since a process model containing noisy loops will have a higher fitness than a process model not showing noise. Therefore, Table 4 shows whether the noisy L1 connection is present in the mined process model. Analogous results (omitted here for the sake of brevity and their straightforward nature) were obtained for L2 loops.

325

It is clear that our newly defined dependency measures for short loops are more selective in terms of presenting them in the mined process model, which increases the validity of the model.

In order to evaluate the validity of the semantic process model, the fitness measure is used. Although fitness is related to completeness, it can also be used here. Namely, the fitness of the semantic process model generated by the Heuristics Miner will be 0 since the generated model is syntactically incorrect and consequently the model will not be able to mine any trace present in the event log. The goal is thus to show that the semantic process models generated by the Updated Heuristics Miner have a value close or equal to 1.

330

To ascertain the validity of the semantic process model generated by the Updated Heuristics Miner, different theoretical cases are investigated. The following cases concerning semantics are possible in an event log: sequential dependencies, XOR dependencies and AND dependencies. Three levels of nesting were used for the semantics, since the case is then shown to work for even and odd levels of nesting and consequently all possible event logs can theoretically be mined correctly. The results (not shown here due to space limitations and their straightforward nature) show that the Heuristics Miner in ProM 6.2 has a fitness value of 0.0 for all cases and that the Updated Heuristics Miner has a fitness value of 1.0 for all cases. The Heuristics Miner in ProM 6.2

340

is thus not able to mine semantic dependencies in an event log, while the Updated Heuristics Miner mines the semantics in a valid way.

6.2. Evaluating the completeness of the mined process models

In order to evaluate the completeness of the mined process models, again the fitness measure is used. Caution is in order, however. A high fitness will indeed imply that the mined process model is more complete, but it may also denote a process model containing noisy dependencies and thus overfitting of the event log occurs. The precision measure from Rozinat et al. (2007b) is appropriate for penalizing overfitted models, however, measuring precision is not feasible. Therefore, in our evaluation, we attempt to consider other elements related to precision in order for a valid conclusion to be made concerning the completeness of the Updated Heuristics Miner.

A first evaluation is focused toward making sure important L2 connections are now shown in the mined process model. This is related to completeness, as discussed in Section 4 and Section 5. Fictitious event logs were constructed and Table 5 shows the results for this evaluation. A log named *L2_90_10* denotes an event log of the following form: $\{CABABD^{90}, CAAD^{10}\}$. Even though the L2 connection consists 90% of the event log, the L2 connection will not be shown since one of the activities is also involved in 10% of the loops of length one. From Table 5, it is clear that the fitness of the Updated Heuristics Miner is higher than or equal to the fitness of the Heuristics Miner in all but one case. The reason for this devious case is that the L1 connection is now the most important connection and consequently the L2 relation will not be shown.

	HM	UHM
L1_0_100	no	no
L1_10_90	yes	no
L1_20_80	yes	no
L1_30_70	yes	no
L1_40_60	yes	no
L1_50_50	yes	yes
L1_60_40	yes	yes
L1_70_30	yes	yes
L1_80_20	yes	yes
L1_90_10	yes	yes
L1_100_0	yes	yes

Table 4: Fitness values for mining an L1 loop with the Heuristics Miner and the Updated Heuristics Miner.

	HM	UHM
L2_0_100	1.0	1.0
L2_10_90	0.9881	0.9881
L2_20_80	0.9773	1.0
L2_30_70	0.9674	1.0
L2_40_60	0.9583	0.9375
L2_50_50	0.95	0.95
L2_60_40	0.9423	0.9615
L2_70_30	0.9352	0.9722
L2_80_20	0.9286	0.9821
L2_90_10	0.9224	0.9914
L2_100_0	1.0	1.0

Table 5: Fitness values for mining an L2 loop in the presence of an L1 loop with the Heuristics Miner and the Updated Heuristics Miner.

In order to conclude whether the Updated Heuristics Miner mines process models with a higher quality than the Heuristics Miner in ProM 6.2, the next evaluation focuses on real-life event logs.

The event logs used for this evaluation are the event log of *De Lijn* and two other real-life event logs made publicly available by (ProcessMiningGroup). The event log of *De Lijn* consists of events related to the customer-complaint process of the company. The second event log *financial log* is related to a fraud detection process and the *volvo* log is an event log of an incident management system of the company *Volvo*.

The fitness values and their constituting variables for these event logs mined by the Heuristics Miner in ProM 6.2 are shown in Table 6. The results for the process models generated by the Updated Heuristics Miner are shown in Table 7. The number of correctly parsed traces c has increased and the number of missing and remaining activities (m and r) has decreased, which means that the mined process models “fit” reality more than the process models mined by the Heuristics Miner in ProM 6.2. An improvement in these values is not the only important aspect, since reaching a fitness value of 1 may imply overfitting of the event logs. The presence of frequent behavior (i.e. dependencies which occur relatively often in an event log, e.g. more than 10%) and the clarity of the mined process models must also be taken into account.

Table 6 and Table 7 also show the percentage of frequent behavior present in the mined process models. It is clear that for two event logs, the percentage of most frequent behavior present in the model is higher for the Updated Heuristics Miner than for the Heuristics Miner. In the other case the percentage does not differ largely and the relative difference is smaller than the large improvement gained for the other two event logs.

When looking at the process model mined by the Updated Heuristics Miner in Figure 5 for *De Lijn*, it is obvious that a clear and less overfitted spaghetti-model is obtained than the process model obtained by the Heuristics Miner in Figure 4. The Updated Heuristics Miner shows all relevant behavior (see Table 7) in the model and is thus more complete.

Figure 6 shows the mined process model of the financial log mined by the Heuristics Miner in ProM 6.2. Comparing this mined process model in terms of clarity to the one generated by the Updated Heuristics Miner, shown in Figure 7, and considering the fact that both process models have the same number of connections, leads to the conclusion that the process model is not deteriorated into a spaghetti-model by an increasing fitness value.

Figure 8 shows the resulting process model for an event log of the incident management system of *Volvo*, mined by the Heuristics Miner algorithm in ProM 6.2. The process model resulting from the same event log by the Updated Heuristics Miner algorithm is shown in Figure 9. It is apparent from Figure 8, that a lot of loops of length one are shown in the process model, even if their frequency is not very high relative to the other frequencies of an activity. Thus, this process model illustrates the problem with the L1 loop dependency measure. In contrast, the Updated Heuristics Miner algorithm only shows the most significant loops. Moreover, note that an important connection is now visible in the process model of the Updated Heuristics Miner which was not visible before,

i.e. the L2 loop between the activities *Accepted in progress* and *Queued Awaiting assignment*. This L2 loop is present in approximately 15.66% of the traces and is part of the most occurring case variants in the event log.

6.3. Evaluation based on cost, effort and time

Obviously, since ProM is open-source, there is no higher cost in using the Updated Heuristics Miner with respect to the Heuristics Miner.

When evaluating the Updated Heuristics Miner based on effort (i.e. the number of mouse clicks a user has to perform), it does not differ much from the implemented Heuristics Miner in ProM. A checkbox “Use updated Heuristics Miner” has been added to the user interface.

Evaluating the Updated Heuristics Miner based on time, resulted in no difference in execution time, which illustrated that no trade-off has to be made between the quality of the mined process model and execution time. The event logs used to compare the execution times are on the one hand conceptual event logs and on the other hand real-life event logs, made available by ProcessMiningGroup (2013).

6.4. Conclusion of the evaluation and possible threats to validity

Considering the objectives set out in Section 3, we can now conclude from the evaluation that the three objectives have been satisfied.

The first objective stated that the validity of the mined process models has to be ensured in the developed artifact. First of all, the new short-loops dependency measures were shown to result in more valid results. The old measures namely resulted in a process model showing noisy short loops and thus in a less valid model for the user. An inherent problem with heuristics is that there is no right answer, so assuming ours is ‘the’ answer would not be a correct conclusion. We believe however, that our dependency heuristic comes closer to a good dependency measure for short loops than the measure from the Heuristics Miner theory. A drawback to our newly defined dependency measures might be that loops are now more easily excluded from the process model. However, we believe it is better to show only relevant connections instead of also showing noisy connections.

Second, the validity of the semantic process models was evaluated by mining fictitious event logs with different levels of semantics. From the construction of these event logs and the results obtained, it can be deduced that

	Heuristics Miner in ProM 6.2							
	<i>c</i>	<i>t</i>	<i>PM</i>	<i>m</i>	<i>r</i>	<i>e</i>	<i>CPM</i>	% of frequent behavior present
<i>De Lijn</i>	6614	16045	0.4122	19850	239	67841	0.8519	31.62%
financial log	6133	13087	0.4686	86057	383	262200	0.8352	97.80%
<i>Volvo</i>	2405	7554	0.3184	30218	699	65533	0.7641	64.68%

Table 6: Fitness values corresponding to the Heuristics Miner algorithm in ProM 6.2 for three real-life event logs.

	Updated Heuristics Miner							
	c	t	PM	m	r	e	CPM	% of frequent behavior present
<i>De Lijn</i>	14451	16045	0.9007	2024	140	67841	0.9841	100%
financial log	6408	13087	0.4895	22529	383	262200	0.9563	93.34%
<i>Volvo</i>	4327	7554	0.5728	8900	225	65533	0.9304	96.48%

Table 7: Fitness values corresponding to the Updated Heuristics Miner for three real-life event logs.

Figure 4: Process model of *De Lijn* mined by the Heuristics Miner in ProM 6.2.

Event logs	Heuristics Miner in ProM 6.2	Updated Heuristics Miner
exercise1	0.011	0.01
exercise2	0.007	0.008
exercise3	0.023	0.022
exercise4	0.015	0.015
exercise5	0.12	0.137
exercise6	0.193	0.056
repairExample	0.352	0.324
repairExample2	0.303	0.315
<i>De Lijn</i>	1.971	1.921
financial log	7.192	7.2
<i>Volvo</i>	2.957	2.877

Table 8: Comparison of the execution times of the Heuristics Miner in ProM 6.2 and the Updated Heuristics Miner.

Figure 5: Process model of *De Lijn* mined by the Updated Heuristics Miner.

Figure 6: Process model of the financial log mined by the Heuristics Miner in ProM 6.2.

Figure 7: Process model of the financial log mined by the Updated Heuristics Miner.

Figure 8: Process model of the event log of *volvo* mined by the Heuristics Miner in ProM 6.2.

Figure 9: Process model of the event log of *volvo* mined by the Updated Heuristics Miner.

the Updated Heuristics Miner now presents a correct semantic process model. A drawback, however, is that this evaluation could not be performed on real-life event logs since ProM does not calculate the fitness for semantic process models and it is not feasible for real-life event logs to calculate manually. The absence of a fitness measure for semantic process models is a disadvantage and an indication that the visualization of semantic process models is a point of improvement in ProM.

The second objective concerns the completeness of the mined process models in ProM. Care should be taken when evaluating completeness. More specifically, the goal is not to include all behavior in the mined process model, which would indeed result in an overfitted model, but to find a trade-off between specialization and generalization. Therefore, we evaluated not only the fitness, but also investigated the presence of frequent behavior. Our findings imply that the Updated Heuristics Miner indeed results in more complete process models (i.e. containing more of the frequent behavior present in the event log) without generating overfitted models. The last columns in Table 6 and Table 7 clearly show that frequent behavior is now present in the event log without overfitting the data, as can be witnessed from the mined process models.

A threat to the validity of our evaluation of completeness lies in the fact that since so little real-life event logs are publicly available, it is not straightforward to perform an extensive evaluation of a process discovery algorithm. Every event log has its own characteristics and a certain process discovery algorithm may or may not be able to deal with these specific characteristics. Moreover, the lack of a clear and structured evaluation framework with accompanying metrics for process discovery techniques makes it difficult to perform an objective evaluation. The lack thereof is an opportunity for further research. A comprehensive and systematic attempt towards the development of such a framework has been made by De Weerd et al. (2012). However, in order for clear-cut conclusions concerning the performance of process discovery techniques, the process mining community

must also make an effort to collect more divergent real-life event logs. This would contribute to process mining research since existing techniques may only then be systematically compared and evaluated.

445

The last objective concerning the cost, the execution time and the effort respectively, are also clearly reached.

7. Conclusion

In this design science paper, we have improved the validity and the completeness of the Heuristics Miner in ProM, a well-known and popular process discovery tool in both research and practice. Three structural problems in the Heuristics Miner algorithm itself were found throughout this work, along with two implementation problems in the ProM 6.2 Heuristics Miner tool.

One structural problem was related to the validity of the mined process model. The definition of dependency measures for short loops resulted in the inclusion of noisy loops in the models. After closer investigation of these measures, it was found that these were counter-intuitive in relation to the intrinsic meaning of these loops. They were changed from being pure frequency measures to dependency measures taking into account their relative importance.

The other structural problems were related to completeness and more specifically to the use of the all-tasks-connected heuristic. This heuristic makes a connection between each event and its predecessor and successor, convinced that all events are connected. However, since other strong connections may exist, the relative-to-best threshold is used. Yet, the Heuristics Miner algorithm does not use this threshold in an intuitive way, i.e. it makes use of another static threshold value which is not in any way related to the relative-to-best threshold. A last structural problem consisted in the exclusion of loops from the all-tasks-connected heuristic. Since this heuristic allows for a partial independence from static threshold values, the Updated Heuristics Miner includes them in the heuristic. These two alterations result in a more intuitive use of threshold values. Therefore, the Updated Heuristics Miner becomes more accessible for non-experienced or first-time users.

We must note that we do not state to have found ‘the’ ultimate solution to this problem; heuristics are intrinsically not optimal. We do believe, however, that these new definitions and assumptions are more intuitive as well as more accurate. This alteration moreover stresses the need for rigid developing and evaluation methods for process discovery techniques. The most important contribution of our work, therefore consists of showing to the community the high need for data and evaluation methods.

Another problem found in the implementation of the Heuristics Miner in ProM 6.2 concerned the semantic process model. This model was not similar to the semantic process models defined in theory. Also, when looking more closely how they were built, it was obvious that these were built in an arbitrary way. In evaluating the solution to this problem, we found that the fitness value could not be calculated using ProM. Therefore, for more complex real-life event logs, this solution could not be evaluated. We however tried to theoretically prove that these real-life event logs can now be semantically mined. This gap between theoretical and practical evaluation

of mined process models, which makes full evaluation of the process models not possible, could be an opportunity for further research.

The relevance of the problems tackled in this paper is obvious when considering the popularity of the Heuristics Miner algorithm in both theory and practice. It is not acceptable for such a technique to lack in validity and completeness. The practical implications of this paper are thus straightforward, i.e. practitioners can now use the Updated Heuristics Miner trusting in the fact that the mined process models now show complete and valid results. The theoretical contributions of this work are less obvious, but nevertheless important. At first sight, this work only shows an improvement to a popular and widely used technique and thus proves the practical relevance. However, two important but indirect theoretical consequences result from this improvement.

First of all, there is an obvious need for an overall developing and evaluation framework for process discovery techniques in the process mining community. Researchers developing a technique in this area are left to trust upon the quality of their algorithm by evaluating it on fictitious event logs and a very small number of real life event logs. Every event log, however, shows different characteristics that may not be accounted for by the developed technique. And thus, using the technique on new available event logs might result in invalid and incomplete mined process models. We strongly advocate further research to focus on the development of such a developing and evaluation framework instead of focusing on the development of new and improved process discovery techniques. Also, a collection of event logs with divergent characteristics should be collected in order for researchers to be able to thoroughly evaluate their process discovery techniques. Then, when such a collection of event logs is present in the community, a benchmarking study can be conducted for the different existing process discovery techniques. This would highly benefit the process mining community since systematically evaluating and comparing existing techniques may lead to the construction of even better techniques.

The second and main theoretical contribution lies in the fact that the Heuristics Miner technique has been used in previous scientific research and these results have contributed to the existing body of knowledge. The validity and completeness of these results, however, can not be guaranteed as can be shown from this work. The improvement of the quality of a research instrument, in this case the Heuristics Miner, will presumably result in an improvement of the quality of the research that uses this instrument.

The contribution of this paper to the research community thus constitutes the improvement of future research related to the Heuristics Miner, the obvious need for a systematic evaluation framework for process mining techniques and the need for an extensive benchmarking study of these techniques.

References

Bibliography

Van der Aalst, W. M. (2011). *Process mining: discovery, conformance and enhancement of business processes*. Springer.

- Van der Aalst, W. M., Andriansyah, A., Alves de Medeiros, A. K., Arcieri, F., Baier, T., Blickle, T., Bose, J. C.,
van den Brand, P., Brandtjen, R., Buijs, J. et al. (2012). Process mining manifesto. In *BPM 2011 Workshops
Proceedings* (pp. 169–194). Springer-Verlag.
- van der Aalst, W. M., Reijers, H. A., Weijters, A. J., van Dongen, B. F., Alves de Medeiros, A., Song, M., &
Verbeek, H. (2007). Business process mining: An industrial application. *Information Systems*, *32*, 713–732.
- Van der Aalst, W. M., & Weijters, A. (2004). Process mining: a research agenda. *Computers in industry*, *53*,
231–244.
- Claes, J., & Poels, G. (2013). Process mining and the prom framework: an exploratory survey. In *Business
Process Management Workshops* (pp. 187–198). Springer.
- De Weerd, J., De Backer, M., Vanthienen, J., & Baesens, B. (2012). A multi-dimensional quality assessment of
state-of-the-art process discovery algorithms using real-life event logs. *Information Systems*, *37*, 654–676.
- Engel, R., van der Aalst, W. M., Zapletal, M., Pichler, C., & Werthner, H. (2012a). Mining inter-organizational
business process models from edi messages: A case study from the automotive sector. In *Advanced Information
Systems Engineering* (pp. 222–237). Springer.
- Engel, R., van der Aalst, W. M., Zapletal, M., Pichler, C., & Werthner, H. (2012b). Mining inter-organizational
business process models from edi messages: A case study from the automotive sector. In *Advanced Information
Systems Engineering* (pp. 222–237). Springer.
- Fei, H., & Meskens, N. (2008). Discovering patient care process models from event logs. In *8th International
Conference of Modeling and Simulation, MOSIM* (pp. 10–12). Citeseer volume 10.
- Gupta, S. (2007). Workflow and process mining in healthcare. *Master's Thesis, Technische Universiteit Eind-
hoven*, .
- Jans, M., van der Werf, J. M., Lybaert, N., & Vanhoof, K. (2011). A business process mining application for
internal transaction fraud mitigation. *Expert Systems with Applications*, *38*, 13351–13359.
- Krogstie, J., Lindland, O. I., & Sindre, G. (1995). Defining quality aspects for conceptual models. *ISCO, 1995*,
216–231.
- LANGab, M., Bürkle, T., Laumann, S., & Prokosch, H.-U. (2008). Process mining for clinical workflows:
challenges and current limitations. In *EHealth Beyond the Horizon: Get It There: Proceedings of MIE2008
the XXist International Congress of the European Federation for Medical Informatics* (p. 229). IOS Press.
- Mans, R., Schonenberg, H., Leonardi, G., Panzarasa, S., Cavallini, A., Quaglini, S., & van der AALST, W. (2008).
Process mining techniques: an application to stroke care. *Studies in health technology and informatics*, *136*,
573.

- Mans, R., Schonenberg, M., Song, M., Aalst, v. d. W., & Bakker, P. (2009). Application of process mining in healthcare—a case study in a dutch hospital. *Biomedical Engineering Systems and Technologies*, (pp. 425–438). 545
- Peffer, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of management information systems*, *24*, 45–77.
- ProcessMiningGroup (2013). <http://www.processmining.org/>.
- Rebuge, Á., & Ferreira, D. R. (2012). Business process analysis in healthcare environments: A methodology based on process mining. *Information Systems*, *37*, 99–116. 550
- Rozinat, A., De Jong, I., Günther, C., & van der Aalst, W. (2007a). *Process mining of test processes: A case study*. Citeseer.
- Rozinat, A., de Medeiros, A., Günther, C., Weijters, A., & van der Aalst, W. (2008). The need for a process mining evaluation framework in research and practice. In *Business Process Management Workshops* (pp. 84–89). Springer. 555
- Rozinat, A., de Medeiros, A. A., Günther, C. W., Weijters, A., & van der Aalst, W. M. (2007b). Towards an evaluation framework for process mining algorithms. *BPM Center Report BPM-07-06*, *BPMcenter.org*, .
- Wang, D., Ge, J., Hu, H., Luo, B., & Huang, L. (2012). Discovering process models from event multiset. *Expert Systems with Applications*, *39*, 11970–11978.
- Wang, Y., Caron, F., Vanthienen, J., Huang, L., & Guo, Y. (2014). Acquiring logistics process intelligence: methodology and an application for a chinese bulk port. *Expert Systems with Applications*, *41*, 195–209. 560
- Weijters, A., van der Aalst, W. M., & de Medeiros, A. A. (2006). Process mining with the heuristics-miner algorithm. *Technische Universiteit Eindhoven, Tech. Rep. WP*, .
- Weijters, A., & Ribeiro, J. (2011). Flexible heuristics miner (fhm). In *Computational Intelligence and Data Mining (CIDM), 2011 IEEE Symposium on* (pp. 310–317). IEEE. 565